Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG
(ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6)
7th Meeting: Pattaya, Thailand, 7-14 March, 2003

Document: JVT-G050
Filename: JVT-G050d35.doc

Title:	**Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)**
Status:	Approved Output Document of JVT
Purpose:	Text

Author(s) or Contact(s):	Thomas Wiegand Heinrich Hertz Institute (FhG), Einsteinufer 37, D-10587 Berlin, Germany	Tel: Fax: Email:	+49 - 30 - 31002 617 +49 - 30 - 392 72 00 wiegand@hhi.de
	Gary Sullivan Microsoft Corporation One Microsoft Way Redmond, WA 98052 USA	Tel: Fax: Email:	+1 (425) 703-5308 +1 (425) 706-7329 garysull@microsoft.com
Source:	Editor		

This document is an output document to the March 2003 Pattaya meeting of the JVT.

Title page to be provided by ITU-T | ISO/IEC

DRAFT INTERNATIONAL STANDARD
DRAFT ISO/IEC 14496-10 : 2002 (E)
DRAFT ITU-T Rec. H.264 (2002 E)
DRAFT ITU-T RECOMMENDATION

TABLE OF CONTENTS

Foreword .... **xi**
0 Introduction .... **xii**
0.1 Prologue xii
0.2 Purpose xii
0.3 Applications xii
0.4 Profiles and levels xii
0.5 Overview of the design characteristics xiii
0.5.1 Predictive coding xiii
0.5.2 Coding of progressive and interlaced video xiii
0.5.3 Picture partitioning into macroblocks and smaller partitions xiii
0.5.4 Spatial redundancy reduction xiv
0.6 How to read this specification xiv
1 Scope **1**
2 Normative references **1**
3 Definitions **1**
4 Abbreviations **7**
5 Conventions **8**
5.1 Arithmetic operators 8
5.2 Logical operators 8
5.3 Relational operators 9

5.4 Bit-wise operators 9
5.5 Assignment operators 9
5.6 Range notation 9
5.7 Mathematical functions 9
5.8 Variables, syntax elements, and tables 10
5.9 Text description of logical operations 11
5.10 Processes 12
6 Source, coded, decoded, output data formats, scanning processes, and neighbouring relationships 12
6.1 Bitstream formats 12
6.2 Source, decoded, and output picture formats 12
6.3 Spatial subdivision of pictures and slices 14
6.4 Inverse scanning processes and derivation processes for neighbours 15
6.4.1 Inverse macroblock scanning process 15
6.4.2 Inverse macroblock partition and sub-macroblock partition scanning process 15
6.4.2.1 Inverse macroblock partition scanning process 16
6.4.2.2 Inverse sub-macroblock partition scanning process 16
6.4.3 Inverse 4x4 luma block scanning process 17
6.4.4 Derivation process of the availability for macroblock addresses 17
6.4.5 Derivation process for neighbouring macroblock addresses and their availability 17
6.4.6 Derivation process for neighbouring macroblock addresses and their availability in MBAFF frames 18
6.4.7 Derivation processes for neighbouring macroblocks, blocks, and partitions 19
6.4.7.1 Derivation process for neighbouring macroblocks 19
6.4.7.2 Derivation process for neighbouring 8x8 luma block 20
6.4.7.3 Derivation process for neighbouring 4x4 luma blocks 20
6.4.7.4 Derivation process for neighbouring 4x4 chroma blocks 21
6.4.7.5 Derivation process for neighbouring partitions 21
6.4.8 Derivation process for neighbouring locations 22
6.4.8.1 Specification for neighbouring luma locations in fields and non-MBAFF frames 22
6.4.8.2 Specification for neighbouring luma locations in MBAFF frames 23
7 Syntax and semantics 25
7.1 Method of describing syntax in tabular form 25
7.2 Specification of syntax functions, categories, and descriptors 26
7.3 Syntax in tabular form 28
7.3.1 NAL unit syntax 28
7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax 29
7.3.2.1 Sequence parameter set RBSP syntax 29
7.3.2.2 Picture parameter set RBSP syntax 30
7.3.2.3 Supplemental enhancement information RBSP syntax 31
7.3.2.3.1 Supplemental enhancement information message syntax 31
7.3.2.4 Access unit delimiter RBSP syntax 31
7.3.2.5 End of sequence RBSP syntax 31
7.3.2.6 End of stream RBSP syntax 32
7.3.2.7 Filler data RBSP syntax 32
7.3.2.8 Slice layer without partitioning RBSP syntax 32
7.3.2.9 Slice data partition RBSP syntax 32
7.3.2.9.1 Slice data partition A RBSP syntax 32
7.3.2.9.2 Slice data partition B RBSP syntax 32
7.3.2.9.3 Slice data partition C RBSP syntax 33
7.3.2.10 RBSP slice trailing bits syntax 33
7.3.2.11 RBSP trailing bits syntax 33
7.3.3 Slice header syntax 34
7.3.3.1 Reference picture list reordering syntax 35
7.3.3.2 Prediction weight table syntax 36
7.3.3.3 Decoded reference picture marking syntax 37
7.3.4 Slice data syntax 38
7.3.5 Macroblock layer syntax 39
7.3.5.1 Macroblock prediction syntax 40
7.3.5.2 Sub-macroblock prediction syntax 41
7.3.5.3 Residual data syntax 42
7.3.5.3.1 Residual block CAVLC syntax 43
7.3.5.3.2 Residual block CABAC syntax 44
7.4 Semantics 45
7.4.1 NAL unit semantics 45

7.4.1.1 Encapsulation of an SODB within an RBSP (informative) 47
7.4.1.2 Order of NAL units and association to coded pictures, access units, and video sequences 48
7.4.1.2.1 Order of sequence and picture parameter set RBSPs and their activation 48
7.4.1.2.2 Order of access units and association to coded video sequences 48
7.4.1.2.3 Order of NAL units and coded pictures and association to access units 49
7.4.1.2.4 Detection of the first VCL NAL unit of a primary coded picture 50
7.4.1.2.5 Order of VCL NAL units and association to coded pictures 50
7.4.2 Raw byte sequence payloads and RBSP trailing bits semantics 51
7.4.2.1 Sequence parameter set RBSP semantics 51
7.4.2.2 Picture parameter set RBSP semantics 53
7.4.2.3 Supplemental enhancement information RBSP semantics 55
7.4.2.3.1 Supplemental enhancement information message semantics 55
7.4.2.4 Access unit delimiter RBSP semantics 55
7.4.2.5 End of sequence RBSP semantics 56
7.4.2.6 End of stream RBSP semantics 56
7.4.2.7 Filler data RBSP semantics 56
7.4.2.8 Slice layer without partitioning RBSP semantics 56
7.4.2.9 Slice data partition RBSP semantics 56
7.4.2.9.1 Slice data partition A RBSP semantics 56
7.4.2.9.2 Slice data partition B RBSP semantics 56
7.4.2.9.3 Slice data partition C RBSP semantics 57
7.4.2.10 RBSP slice trailing bits semantics 57
7.4.2.11 RBSP trailing bits semantics 57
7.4.3 Slice header semantics 57
7.4.3.1 Reference picture list reordering semantics 62
7.4.3.2 Prediction weight table semantics 63
7.4.3.3 Decoded reference picture marking semantics 64
7.4.4 Slice data semantics 66
7.4.5 Macroblock layer semantics 66
7.4.5.1 Macroblock prediction semantics 73
7.4.5.2 Sub-macroblock prediction semantics 73
7.4.5.3 Residual data semantics 76
7.4.5.3.1 Residual block CAVLC semantics 76
7.4.5.3.2 Residual block CABAC semantics 76

8 Decoding process 77
8.1 NAL unit decoding process 78
8.2 Slice decoding process 78
8.2.1 Decoding process for picture order count 78
8.2.1.1 Decoding process for picture order count type 0 80
8.2.1.2 Decoding process for picture order count type 1 80
8.2.1.3 Decoding process for picture order count type 2 81
8.2.2 Decoding process for macroblock to slice group map 82
8.2.2.1 Specification for interleaved slice group map type 83
8.2.2.2 Specification for dispersed slice group map type 83
8.2.2.3 Specification for foreground with left-over slice group map type 83
8.2.2.4 Specification for box-out slice group map types 84
8.2.2.5 Specification for raster scan slice group map types 84
8.2.2.6 Specification for wipe slice group map types 85
8.2.2.7 Specification for explicit slice group map type 85
8.2.2.8 Specification for conversion of map unit to slice group map to macroblock to slice group map 85
8.2.3 Decoding process for slice data partitioning 85
8.2.4 Decoding process for reference picture lists construction 86
8.2.4.1 Decoding process for picture numbers 86
8.2.4.2 Initialisation process for reference picture lists 87
8.2.4.2.1 Initialisation process for the reference picture list for P and SP slices in frames 87
8.2.4.2.2 Initialisation process for the reference picture list for P and SP slices in fields 88
8.2.4.2.3 Initialisation process for reference picture lists for B slices in frames 88
8.2.4.2.4 Initialisation process for reference picture lists for B slices in fields 89
8.2.4.2.5 Initialisation process for reference picture lists in fields 90
8.2.4.3 Reordering process for reference picture lists 90
8.2.4.3.1 Reordering process of reference picture lists for short-term pictures 91
8.2.4.3.2 Reordering process of reference picture lists for long-term pictures 92
8.2.5 Decoded reference picture marking process 92

8.2.5.1 Sequence of operations for decoded reference picture marking process 92
8.2.5.2 Decoding process for gaps in frame_num.. 93
8.2.5.3 Sliding window decoded reference picture marking process.. 93
8.2.5.4 Adaptive memory control decoded reference picture marking process 93
8.2.5.4.1 Marking process of a short-term picture as “unused for reference”.. 94
8.2.5.4.2 Marking process of a long-term picture as “unused for reference”... 94
8.2.5.4.3 Assignment process of a LongTermFrameIdx to a short-term reference picture 94
8.2.5.4.4 Decoding process for MaxLongTermFrameIdx.. 95
8.2.5.4.5 Marking process of all reference pictures as “unused for reference” and setting MaxLongTermFrameIdx to “no long-term frame indices”... 95
8.2.5.4.6 Process for assigning a long-term frame index to the current picture 95
8.3 Intra prediction process.. 95
8.3.1 Intra_4x4 prediction process for luma samples 96
8.3.1.1 Derivation process for the Intra4x4PredMode.. 96
8.3.1.2 Intra_4x4 sample prediction 98
8.3.1.2.1 Specification of Intra_4x4_Vertical prediction mode 98
8.3.1.2.2 Specification of Intra_4x4_Horizontal prediction mode.. 98
8.3.1.2.3 Specification of Intra_4x4_DC prediction mode 99
8.3.1.2.4 Specification of Intra_4x4_Diagonal_Down_Left prediction mode 99
8.3.1.2.5 Specification of Intra_4x4_Diagonal_Down_Right prediction mode.. 99
8.3.1.2.6 Specification of Intra_4x4_Vertical_Right prediction mode 100
8.3.1.2.7 Specification of Intra_4x4_Horizontal_Down prediction mode 100
8.3.1.2.8 Specification of Intra_4x4_Vertical_Left prediction mode 100
8.3.1.2.9 Specification of Intra_4x4_Horizontal_Up prediction mode 101
8.3.2 Intra_16x16 prediction process for luma samples 101
8.3.2.1 Specification of Intra_16x16_Vertical prediction mode.. 102
8.3.2.2 Specification of Intra_16x16_Horizontal prediction mode... 102
8.3.2.3 Specification of Intra_16x16_DC prediction mode 102
8.3.2.4 Specification of Intra_16x16_Plane prediction mode... 102
8.3.3 Intra prediction process for chroma samples 103
8.3.3.1 Specification of Intra_Chroma_DC prediction mode 104
8.3.3.2 Specification of Intra_Chroma_Horizontal prediction mode.. 105
8.3.3.3 Specification of Intra_Chroma_Vertical prediction mode 105
8.3.3.4 Specification of Intra_Chroma_Plane prediction mode... 105
8.3.4 Sample construction process for I_PCM macroblocks 106
8.4 Inter prediction process.. 106
8.4.1 Derivation process for motion vector components and reference indices...................................... 108
8.4.1.1 Derivation process for luma motion vectors for skipped macroblocks in P and SP slices.............. 108
8.4.1.2 Derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8 109
8.4.1.2.1 Derivation process for the co-located 4x4 sub-macroblock partitions 109
8.4.1.2.2 Derivation process for spatial direct luma motion vector and reference index prediction mode ... 112
8.4.1.2.3 Derivation process for temporal direct luma motion vector and reference index prediction mode 113
8.4.1.3 Derivation process for luma motion vector prediction 115
8.4.1.3.1 Derivation process for median luma motion vector prediction 116
8.4.1.3.2 Derivation process for motion data of neighbouring partitions.. 116
8.4.1.4 Derivation process for chroma motion vectors 117
8.4.2 Decoding process for Inter prediction samples.. 118
8.4.2.1 Reference picture selection process 118
8.4.2.2 Fractional sample interpolation process.. 119
8.4.2.2.1 Luma sample interpolation process.. 120
8.4.2.2.2 Chroma sample interpolation process 122
8.4.2.3 Weighted sample prediction process 123
8.4.2.3.1 Default weighted sample prediction process... 124
8.4.2.3.2 Weighted sample prediction process.. 124
8.5 Transform coefficient decoding process and picture construction process prior to deblocking filter process.. 126
8.5.1 Specification of transform decoding process for residual blocks.. 126
8.5.2 Specification of transform decoding process for luma samples of Intra_16x16 macroblock prediction mode 127
8.5.3 Specification of transform decoding process for chroma samples... 128
8.5.4 Inverse scanning process for transform coefficients.. 129
8.5.5 Derivation process for the quantisation parameters and scaling function....................................... 129
8.5.6 Scaling and transformation process for luma DC transform coefficients for Intra_16x16 macroblock type 130

8.5.7 Scaling and transformation process for chroma DC transform coefficients 131
8.5.8 Scaling and transformation process for residual 4x4 blocks 131
8.5.9 Picture construction process prior to deblocking filter process 133
8.6 Decoding process for P macroblocks in SP slices or SI macroblocks 133
8.6.1 SP decoding process for non-switching pictures 134
8.6.1.1 Luma transform coefficient decoding process 134
8.6.1.2 Chroma transform coefficient decoding process 135
8.6.2 SP and SI slice decoding process for switching pictures 136
8.6.2.1 Luma transform coefficient decoding process 137
8.6.2.2 Chroma transform coefficient decoding process 137
8.7 Deblocking filter process 138
8.7.1 Filtering process for block edges 141
8.7.2 Filtering process for a set of samples across a horizontal or vertical block edge 142
8.7.2.1 Derivation process for the luma content dependent boundary filtering strength 143
8.7.2.2 Derivation process for the thresholds for each block edge 144
8.7.2.3 Filtering process for edges with bS less than 4 145
8.7.2.4 Filtering process for edges for bS equal to 4 146

9 Parsing process 147
9.1 Parsing process for Exp-Golomb codes 147
9.1.1 Mapping process for signed Exp-Golomb codes 149
9.1.2 Mapping process for coded block pattern 149
9.2 CAVLC parsing process for transform coefficient levels 151
9.2.1 Parsing process for total number of transform coefficient levels and trailing ones 151
9.2.2 Parsing process for level information 154
9.2.3 Parsing process for run information 156
9.2.4 Combining level and run information 158
9.3 CABAC parsing process for slice data 159
9.3.1 Initialisation process 160
9.3.1.1 Initialisation process for context variables 160
9.3.1.2 Initialisation process for the arithmetic decoding engine 170
9.3.2 Binarization process 170
9.3.2.1 Unary (U) binarization process 172
9.3.2.2 Truncated unary (TU) binarization process 172
9.3.2.3 Concatenated unary/ k-th order Exp-Golomb (UEGk) binarization process 172
9.3.2.4 Fixed-length (FL) binarization process 173
9.3.2.5 Binarization process for macroblock type and sub-macroblock type 173
9.3.2.6 Binarization process for coded block pattern 176
9.3.2.7 Binarization process for mb_qp_delta 176
9.3.3 Decoding process flow 176
9.3.3.1 Derivation process for ctxIdx 177
9.3.3.1.1 Assignment process of ctxIdxInc using neighbouring syntax elements 179
9.3.3.1.1.1 Derivation process of ctxIdxInc for the syntax element mb_skip_flag 179
9.3.3.1.1.2 Derivation process of ctxIdxInc for the syntax element mb_field_decoding_flag 179
9.3.3.1.1.3 Derivation process of ctxIdxInc for the syntax element mb_type 180
9.3.3.1.1.4 Derivation process of ctxIdxInc for the syntax element coded_block_pattern 180
9.3.3.1.1.5 Derivation process of ctxIdxInc for the syntax element mb_qp_delta 181
9.3.3.1.1.6 Derivation process of ctxIdxInc for the syntax elements ref_idx_l0 and ref_idx_l1 181
9.3.3.1.1.7 Derivation process of ctxIdxInc for the syntax elements mvd_l0 and mvd_l1 182
9.3.3.1.1.8 Derivation process of ctxIdxInc for the syntax element intra_chroma_pred_mode 182
9.3.3.1.1.9 Derivation process of ctxIdxInc for the syntax element coded_block_flag 183
9.3.3.1.2 Assignment process of ctxIdxInc using prior decoded bin values 184
9.3.3.1.3 Assignment process of ctxIdxInc for syntax elements significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1 184
9.3.3.2 Arithmetic decoding process 185
9.3.3.2.1 Arithmetic decoding process for a binary decision 186
9.3.3.2.1.1 State transition process 187
9.3.3.2.2 Renormalization process in the arithmetic decoding engine 189
9.3.3.2.3 Bypass decoding process for binary decisions 189
9.3.3.2.4 Decoding process for binary decisions before termination 190
9.3.4 Arithmetic encoding process (informative) 191
9.3.4.1 Initialisation process for the arithmetic encoding engine (informative) 191
9.3.4.2 Encoding process for a binary decision (informative) 191
9.3.4.3 Renormalization process in the arithmetic encoding engine (informative) 192

9.3.4.4 Bypass encoding process for binary decisions (informative) 193
9.3.4.5 Encoding process for a binary decision before termination (informative) 194
9.3.4.6 Byte stuffing process (informative) 195

Annex A Profiles and levels 196

A.1 Requirements on video decoder capability 196
A.2 Profiles 196
A.2.1 Baseline profile 196
A.2.2 Main profile 196
A.2.3 Extended profile 197
A.3 Levels 197
A.3.1 Profile-independent level limits 197
A.3.2 Profile-specific level limits 199
A.3.2.1 Baseline profile limits 200
A.3.2.2 Main profile limits 200
A.3.2.3 Extended Profile Limits 201
A.3.3 Effect of level limits on frame rate (informative) 202

Annex B Byte stream format 203

B.1 Byte stream NAL unit syntax and semantics 204
B.1.1 Byte stream NAL unit syntax 204
B.1.2 Byte stream NAL unit semantics 204
B.2 Byte stream NAL unit decoding process 204
B.3 Decoder byte-alignment recovery (informative) 205

Annex C Hypothetical reference decoder 205

C.1 Operation of coded picture buffer (CPB) 208
C.1.1 Timing of bitstream arrival 208
C.1.2 Timing of coded picture removal 209
C.2 Operation of the decoded picture buffer (DPB) 209
C.2.1 Decoding of gaps in frame_num and storage of "non-existing" frames 209
C.2.2 Picture decoding and output 210
C.2.3 Removal of pictures from the DPB before possible insertion of the current picture 210
C.2.4 Current decoded picture marking and storage 210
C.2.4.1 Marking and storage of a reference decoded picture into the DPB 210
C.2.4.2 Storage of a non-reference picture into the DPB 211
C.3 Bitstream conformance 211
C.4 Decoder conformance 212
C.4.1 Operation of the output order DPB 213
C.4.2 Decoding of gaps in frame_num and storage of "non-existing" pictures 213
C.4.3 Picture decoding 213
C.4.4 Removal of pictures from the DPB before possible insertion of the current picture 213
C.4.5 Current decoded picture marking and storage 214
C.4.5.1 Storage of picture order counts for the decoded picture 214
C.4.5.2 Storage and marking of a reference decoded picture into the DPB 214
C.4.5.3 Storage and marking of a non-reference decoded picture into the DPB 214
C.4.5.4 "Bumping" process 214

Annex D Supplemental enhancement information 215

D.1 SEI payload syntax 216
D.1.1 Buffering period SEI message syntax 217
D.1.2 Picture timing SEI message syntax 217
D.1.3 Pan-scan rectangle SEI message syntax 218
D.1.4 Filler payload SEI message syntax 218
D.1.5 User data registered by ITU-T Recommendation T.35 SEI message syntax 219
D.1.6 User data unregistered SEI message syntax 219
D.1.7 Recovery point SEI message syntax 219
D.1.8 Decoded reference picture marking repetition SEI message syntax 219
D.1.9 Spare picture SEI message syntax 220
D.1.10 Scene information SEI message syntax 220
D.1.11 Sub-sequence information SEI message syntax 221
D.1.12 Sub-sequence layer characteristics SEI message syntax 221
D.1.13 Sub-sequence characteristics SEI message syntax 221
D.1.14 Full-frame freeze SEI message syntax 222
D.1.15 Full-frame freeze release SEI message syntax 222
D.1.16 Full-frame snapshot SEI message syntax 222

D.1.17 Progressive refinement segment start SEI message syntax 222
D.1.18 Progressive refinement segment end SEI message syntax 222
D.1.19 Motion-constrained slice group set SEI message syntax 222
D.1.20 Reserved SEI message syntax 223
D.2 SEI payload semantics 223
D.2.1 Buffering period SEI message semantics 223
D.2.2 Picture timing SEI message semantics 223
D.2.3 Pan-scan rectangle SEI message semantics 226
D.2.4 Filler payload SEI message semantics 228
D.2.5 User data registered by ITU-T Recommendation T.35 SEI message semantics 228
D.2.6 User data unregistered SEI message semantics 228
D.2.7 Recovery point SEI message semantics 228
D.2.8 Decoded reference picture marking repetition SEI message semantics 230
D.2.9 Spare picture SEI message semantics 230
D.2.10 Scene information SEI message semantics 231
D.2.11 Sub-sequence information SEI message semantics 233
D.2.12 Sub-sequence layer characteristics SEI message semantics 234
D.2.13 Sub-sequence characteristics SEI message semantics 235
D.2.14 Full-frame freeze SEI message semantics 236
D.2.15 Full-frame freeze release SEI message semantics 236
D.2.16 Full-frame snapshot SEI message semantics 237
D.2.17 Progressive refinement segment start SEI message semantics 237
D.2.18 Progressive refinement segment end SEI message semantics 237
D.2.19 Motion-constrained slice group set SEI message semantics 237
D.2.20 Reserved SEI message semantics 238
Annex E Video usability information 238
E.1 VUI syntax 239
E.1.1 VUI parameters syntax 239
E.1.2 HRD parameters syntax 240
E.2 VUI semantics 240
E.2.1 VUI parameters semantics 240
E.2.2 HRD parameters semantics 249

LIST OF FIGURES

Figure 6-1 – Nominal vertical and horizontal locations of 4:2:0 luma and chroma samples in a frame 13

Figure 6-2 – Nominal vertical and horizontal sampling locations of samples top and bottom fields. 14

Figure 6-3 – A picture with 11 by 9 macroblocks that is partitioned into two slices 14

Figure 6-4 – Partitioning of the decoded frame into macroblock pairs. 15

Figure 6-5 – Macroblock partitions, sub-macroblock partitions, macroblock partition scans, and sub-macroblock partition scans. 16

Figure 6-6 – Scan for 4x4 luma blocks. 17

Figure 6-7 – Neighbouring macroblocks for a given macroblock 18

Figure 6-8 – Neighbouring macroblocks for a given macroblock in MBAFF frames 18

Figure 6-9 – Determination of the neighbouring macroblock, blocks, and partitions (informative) 19

Figure 7-1 – The structure of an access unit not containing any NAL units with nal_unit_type equal to 0, 7, 8, or in the range of 12 to 31, inclusive 50

Figure 8-1 – Intra_4x4 prediction mode directions (informative) 97

Figure 8-2 –Example for temporal direct-mode motion vector inference (informative) 115

Figure 8-3 – Directional segmentation prediction (informative) 116

Figure 8-4 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for quarter sample luma interpolation. 121

Figure 8-5 – Fractional sample position dependent variables in chroma interpolation and surrounding integer position samples A, B, C, and D 123

Figure 8-6 – Assignment of the indices of dcY to luma4x4BlkIdx. 127

Figure 8-7 – Assignment of the indices of dcC to chroma4x4BlkIdx. 128

Figure 8-8 – a) Zig-zag scan. b) Field scan 129

Figure 8-9 – Boundaries in a macroblock to be filtered (luma boundaries shown with solid lines and chroma boundaries shown with dashed lines) 139

Figure 8-10 – Convention for describing samples across a 4x4 block horizontal or vertical boundary 142

Figure 9-1 – Illustration of CABAC parsing process for a syntax element SE (informative) 160

Figure 9-2 – Overview of the arithmetic decoding process for a single bin (informative) 186

Figure 9-3 – Flowchart for decoding a decision 187

Figure 9-4 – Flowchart of renormalization 189

Figure 9-5 – Flowchart of bypass decoding process 190

Figure 9-6 – Flowchart of decoding a decision before termination 191

Figure 9-7 – Flowchart for encoding a decision 192

Figure 9-8 – Flowchart of renormalization in the encoder 193

Figure 9-9 – Flowchart of PutBit(B) 193

Figure 9-10 – Flowchart of encoding bypass 194

Figure 9-11 – Flowchart of encoding a decision before termination 195

Figure 9-12 – Flowchart of flushing at termination 195

Figure C-1 – Structure of byte streams and NAL unit streams and HRD conformance points 206

Figure C-2 – HRD buffer model 207

Figure E-1 – Location of chroma samples for top and bottom fields as a function of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field 246

LIST OF TABLES

Table 6-1 – ChromaFormatFactor values 12

Table 6-2 – Specification of input and output assignments for subclauses 6.4.7.1 to 6.4.7.5 19

Table 6-3 – Specification of mbAddrN 23

Table 6-4 - Specification of mbAddrN and yM 24

Table 7-1 – NAL unit type codes 46

Table 7-2 – Meaning of primary_pic_type 56

Table 7-3 – Name association to slice_type 58

Table 7-4 – reordering_of_pic_nums_idc operations for reordering of reference picture lists 63

Table 7-5 – Interpretation of adaptive_ref_pic_marking_mode_flag 64

Table 7-6 – Memory management control operation (memory_management_control_operation) values 65

Table 7-7 – Allowed collective macroblock types for slice_type 67

Table 7-8 – Macroblock types for I slices 68

Table 7-9 – Macroblock type with value 0 for SI slices 69

Table 7-10 – Macroblock type values 0 to 4 for P and SP slices 70

Table 7-11 – Macroblock type values 0 to 22 for B slices 71

Table 7-12 – Specification of CodedBlockPatternChroma values 72

Table 7-13 – Relationship between intra_chroma_pred_mode and spatial prediction modes 73

Table 7-14 – Sub-macroblock types in P macroblocks........74
Table 7-15 – Sub-macroblock types in B macroblocks........75
Table 8-1 – Refined slice group map type........82
Table 8-2 – Specification of Intra4x4PredMode[luma4x4BlkIdx] and associated names........96
Table 8-3 – Specification of Intra16x16PredMode and associated names........102
Table 8-4 – Specification of Intra chroma prediction modes and associated names........103
Table 8-5 – Specification of the variable colPic........109
Table 8-6 – Specification of PicCodingStruct(X)........110
Table 8-7 – Specification of mbAddrCol, yM, and vertMvScale........111
Table 8-8 – Assignment of prediction utilization flags........113
Table 8-9 – Derivation of the vertical component of the chroma vector in field coding mode........118
Table 8-10 – Differential full-sample luma locations........121
Table 8-11 – Assignment of the luma prediction sample predPartLX$_L$[x_L, y_L]........122
Table 8-12 – Specification of mapping of idx to c_{ij} for zig-zag and field scan........129
Table 8-13 – Specification of QP_C as a function of qP_I........130
Table 8-14 – Derivation of indexA and indexB from offset dependent threshold variables α and β........145
Table 8-15 – Value of filter clipping variable t_{C0} as a function of indexA and bS........146
Table 9-1 – Bit strings with “prefix” and “suffix” bits and assignment to codeNum ranges (informative)........148
Table 9-2 – Exp-Golomb bit strings and codeNum in explicit form and used as ue(v) (informative)........148
Table 9-3 – Assignment of syntax element to codeNum for signed Exp-Golomb coded syntax elements se(v)........149
Table 9-4 – Assignment of codeNum to values of coded_block_pattern for macroblock prediction modes........149
Table 9-5 – coeff_token mapping to TotalCoeff(coeff_token) and TrailingOnes(coeff_token)........153
Table 9-6 – Codeword table for level_prefix........156
Table 9-7 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 1 to 7........157
Table 9-8 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 8 to 15........157
Table 9-9 – total_zeros tables for chroma DC 2x2 blocks........158
Table 9-10 – Tables for run_before........158
Table 9-11 – Association of ctxIdx and syntax elements for each slice type in the initialisation process........161
Table 9-12 – Values of variables m and n for ctxIdx from 0 to 10........162
Table 9-13 – Values of variables m and n for ctxIdx from 11 to 23........162
Table 9-14 – Values of variables m and n for ctxIdx from 24 to 39........162
Table 9-15 – Values of variables m and n for ctxIdx from 40 to 53........163
Table 9-16 – Values of variables m and n for ctxIdx from 54 to 59........163
Table 9-17 – Values of variables m and n for ctxIdx from 60 to 69........163
Table 9-18 – Values of variables m and n for ctxIdx from 70 to 104........164
Table 9-19 – Values of variables m and n for ctxIdx from 105 to 165........165
Table 9-20 – Values of variables m and n for ctxIdx from 166 to 226........166
Table 9-21 – Values of variables m and n for ctxIdx from 227 to 275........167
Table 9-22 – Values of variables m and n for ctxIdx from 277 to 337........168
Table 9-23 – Values of variables m and n for ctxIdx from 338 to 398........169

Table 9-24 – Syntax elements and associated types of binarization, maxBinIdxCtx, and ctxIdxOffset........................... 171
Table 9-25 – Bin string of the unary binarization (informative).. 172
Table 9-26 – Binarization for macroblock types in I slices 174
Table 9-27 – Binarization for macroblock types in P, SP, and B slices.. 175
Table 9-28 – Binarization for sub-macroblock types in P, SP, and B slices 176
Table 9-29 – Assignment of ctxIdxInc to binIdx for all ctxIdxOffset values except those related to the syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1..................... 178
Table 9-30 – Assignment of ctxIdxBlockCatOffset to ctxBlockCat for syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1 179
Table 9-31 – Specification of ctxIdxInc for specific values of ctxIdxOffset and binIdx 184
Table 9-32 – Specification of ctxBlockCat for the different blocks 185
Table 9-33 – Specification of rangeTabLPS depending on pStateIdx and qCodIRangeIdx 188
Table 9-34 – State transition table 189
Table A-1 – Level limits... 199
Table A-2 – Baseline profile level limits.. 200
Table A-3 – Main profile level limits 201
Table A-4 – Extended profile level limits 201
Table A-5 – Maximum frame rates (frames per second) for some example frame sizes... 202
Table D-1 – Interpretation of pic_struct 224
Table D-2 – Mapping of ct_type to source picture scan 225
Table D-3 – Definition of counting_type values 225
Table D-4 – scene_transition_type values. 232
Table E-1 – Meaning of sample aspect ratio indicator 241
Table E-2 – Meaning of video_format 242
Table E-3 – Colour primaries 243
Table E-4 – Transfer characteristics 244
Table E-5 – Matrix coefficients... 245
Table E-6 – Divisor for computation of $\Delta t_{fi,dpb}(n)$ 247

Foreword

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardisation Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardising telecommunications on a world-wide basis. The World Telecommunication Standardisation Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups that, in turn, produce Recommendations on these topics. The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1. In some areas of information technology that fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

ISO (the International Organisation for Standardisation) and IEC (the International Electrotechnical Commission) form the specialised system for world-wide standardisation. National Bodies that are members of ISO and IEC participate in the development of International Standards through technical committees established by the respective organisation to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organisations, 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 JTC1. 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.

This Recommendation | International Standard was prepared jointly by ITU-T SG16 Q.6, also known as VCEG (Video Coding Experts Group), and by ISO/IEC JTC1/SC29/WG11, also known as MPEG (Moving Picture Experts Group). VCEG was formed in 1997 to maintain prior ITU-T video coding standards and develop new video coding standard(s) appropriate for a wide range of conversational and non-conversational services. MPEG was formed in 1988 to establish standards for coding of moving pictures and associated audio for various applications such as digital storage media, distribution, and communication.

In this Recommendation | International Standard Annexes A through E contain normative requirements and are an integral part of this Recommendation | International Standard.

0 Introduction

This clause does not form an integral part of this Recommendation | International Standard.

0.1 Prologue

This subclause does not form an integral part of this Recommendation | International Standard.

As the costs for both processing power and memory have reduced, network support for coded video data has diversified, and advances in video coding technology have progressed, the need has arisen for an industry standard for compressed video representation with substantially increased coding efficiency and enhanced robustness to network environments. Toward these ends the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) formed a Joint Video Team (JVT) in 2001 for development of a new Recommendation | International Standard.

0.2 Purpose

This subclause does not form an integral part of this Recommendation | International Standard.

This Recommendation | International Standard was developed in response to the growing need for higher compression of moving pictures for various applications such as videoconferencing, digital storage media, television broadcasting, internet streaming, and communication. It is also designed to enable the use of the coded video representation in a flexible manner for a wide variety of network environments. The use of this Recommendation | International Standard allows motion video to be manipulated as a form of computer data and to be stored on various storage media, transmitted and received over existing and future networks and distributed on existing and future broadcasting channels.

0.3 Applications

This subclause does not form an integral part of this Recommendation | International Standard.

This Recommendation | International Standard is designed to cover a broad range of applications for video content including but not limited to the following:

- CATV Cable TV on optical networks, copper, etc.
- DBS Direct broadcast satellite video services
- DSL Digital subscriber line video services
- DTTB Digital terrestrial television broadcasting
- ISM Interactive storage media (optical disks, etc.)
- MMM Multimedia mailing
- MSPN Multimedia services over packet networks
- RTC Real-time conversational services (videoconferencing, videophone, etc.)
- RVS Remote video surveillance
- SSM Serial storage media (digital VTR, etc.)

0.4 Profiles and levels

This subclause does not form an integral part of this Recommendation | International Standard.

This Recommendation | International Standard is designed to be generic in the sense that it serves a wide range of applications, bit rates, resolutions, qualities, and services. Applications should cover, among other things, digital storage media, television broadcasting and real-time communications. In the course of creating this Specification, various requirements from typical applications have been considered, necessary algorithmic elements have been developed, and these have been integrated into a single syntax. Hence, this Specification will facilitate video data interchange among different applications.

Considering the practicality of implementing the full syntax of this Specification, however, a limited number of subsets of the syntax are also stipulated by means of "profiles" and "levels". These and other related terms are formally defined in clause 3.

A "profile" is a subset of the entire bitstream syntax that is specified by this Recommendation | International Standard. Within the bounds imposed by the syntax of a given profile it is still possible to require a very large variation in the performance of encoders and decoders depending upon the values taken by syntax elements in the bitstream such as the specified size of the decoded pictures. In many applications, it is currently neither practical nor economic to implement a decoder capable of dealing with all hypothetical uses of the syntax within a particular profile.

In order to deal with this problem, "levels" are specified within each profile. A level is a specified set of constraints imposed on values of the syntax elements in the bitstream. These constraints may be simple limits on values. Alternatively they may take the form of constraints on arithmetic combinations of values (e.g. picture width multiplied by picture height multiplied by number of pictures decoded per second).

Coded video content conforming to this Recommendation | International Standard uses a common syntax. In order to achieve a subset of the complete syntax, flags, parameters, and other syntax elements are included in the bitstream that signal the presence or absence of syntactic elements that occur later in the bitstream.

0.5 Overview of the design characteristics

This subclause does not form an integral part of this Recommendation | International Standard.

The coded representation specified in the syntax is designed to enable a high compression capability for a desired image quality. The algorithm is not lossless, as the exact source sample values are typically not preserved through the encoding and decoding processes. A number of techniques may be used to achieve highly efficient compression. The expected encoding algorithm (not specified in this Recommendation | International Standard) selects between inter and intra coding for block-shaped regions of each picture. Inter coding uses motion vectors for block-based inter prediction to exploit temporal statistical dependencies between different pictures. Intra coding uses various spatial prediction modes to exploit spatial statistical dependencies in the source signal for a single picture. Motion vectors and intra prediction modes may be specified for a variety of block sizes in the picture. The prediction residual is then further compressed using a transform to remove spatial correlation inside the transform block before it is quantised, producing an irreversible process that typically discards less important visual information while forming a close approximation to the source samples. Finally, the motion vectors or intra prediction modes are combined with the quantised transform coefficient information and encoded using either variable length codes or arithmetic coding.

0.5.1 Predictive coding

This subclause does not form an integral part of this Recommendation | International Standard.

Because of the conflicting requirements of random access and highly efficient compression, two main coding types are specified. Intra coding is done without reference to other pictures. Intra coding may provide access points to the coded sequence where decoding can begin and continue correctly, but typically also shows only moderate compression efficiency. Inter coding (predictive or bi-predictive) is more efficient using inter prediction of each block of sample values from some previously decoded picture selected by the encoder. In contrast to some other video coding standards, pictures coded using bi-predictive inter prediction may also be used as references for inter coding of other pictures.

The application of the three coding types to pictures in a sequence is flexible, and the order of the decoding process is generally not the same as the order of the source picture capture process in the encoder or the output order from the decoder for display. The choice is left to the encoder and will depend on the requirements of the application. The decoding order is specified such that the decoding of pictures that use inter-picture prediction follows later in decoding order than other pictures that are referenced in the decoding process.

0.5.2 Coding of progressive and interlaced video

This subclause does not form an integral part of this Recommendation | International Standard.

This Recommendation | International Standard specifies a syntax and decoding process for video that originated in either progressive-scan or interlaced-scan form, which may be mixed together in the same sequence. The two fields of an interlaced frame are separated in capture time while the two fields of a progressive frame share the same capture time. Each field may be coded separately or the two fields may be coded together as a frame. Progressive frames are typically coded as a frame. For interlaced video, the encoder can choose between frame coding and field coding. Frame coding or field coding can be adaptively selected on a picture-by-picture basis and also on a more localized basis within a coded frame. Frame coding is typically preferred when the video scene contains significant detail with limited motion. Field coding typically works better when there is fast picture-to-picture motion.

0.5.3 Picture partitioning into macroblocks and smaller partitions

This subclause does not form an integral part of this Recommendation | International Standard.

As in previous video coding Recommendations and International Standards, a macroblock, consisting of a 16x16 block of luma samples and two corresponding blocks of chroma samples, is used as the basic processing unit of the video decoding process.

A macroblock can be further partitioned for inter prediction. The selection of the size of inter prediction partitions is a result of a trade-off between the coding gain provided by using motion compensation with smaller blocks and the quantity of data needed to represent the data for motion compensation. In this Recommendation | International Standard the inter prediction process can form segmentations for motion representation as small as 4x4 luma samples in size, using motion vector accuracy of one-quarter of the luma sample grid spacing displacement. The process for inter prediction of a sample block can also involve the selection of the picture to be used as the reference picture from a number of stored previously-decoded pictures. Motion vectors are encoded differentially with respect to predicted values formed from nearby encoded motion vectors.

Typically, the encoder calculates appropriate motion vectors and other data elements represented in the video data stream. This motion estimation process in the encoder and the selection of whether to use inter prediction for the representation of each region of the video content is not specified in this Recommendation | International Standard.

0.5.4 Spatial redundancy reduction

This subclause does not form an integral part of this Recommendation | International Standard.

Both source pictures and prediction residuals have high spatial redundancy. This Recommendation | International Standard is based on the use of a block-based transform method for spatial redundancy removal. After inter prediction from previously-decoded samples in other pictures or spatial-based prediction from previously-decoded samples within the current picture, the resulting prediction residual is split into 4x4 blocks. These are converted into the transform domain where they are quantised. After quantisation many of the transform coefficients are zero or have low amplitude and can thus be represented with a small amount of encoded data. The processes of transformation and quantisation in the encoder are not specified in this Recommendation | International Standard.

0.6 How to read this specification

This subclause does not form an integral part of this Recommendation | International Standard.

It is suggested that the reader starts with clause 1 (Scope) and moves on to clause 3 (Definitions). Clause 6 should be read for the geometrical relationship of the source, input, and output of the decoder. Clause 7 (Syntax and semantics) specifies the order to parse syntax elements from the bitstream. See subclauses 7.1-7.3 for syntactical order and see subclause 7.4 for semantics; i.e., the scope, restrictions, and conditions that are imposed on the syntax elements. The actual parsing for most syntax elements is specified in clause 9 (Parsing process). Finally, clause 8 (Decoding process) specifies how the syntax elements are mapped into decoded samples. Throughout reading this specification, the reader should refer to clauses 2 (Normative references), 4 (Abbreviations), and 5 (Conventions) as needed. Annexes A through E also form an integral part of this Recommendation | International Standard.

Annex A defines three profiles (Baseline, Main, and Extended), each being tailored to certain application domains, and defines the so-called levels of the profiles. Annex B specifies syntax and semantics of a byte stream format for delivery of coded video as an ordered stream of bytes. Annex C specifies the hypothetical reference decoder and its use to check bitstream and decoder conformance. Annex D specifies syntax and semantics for supplemental enhancement information message payloads. Finally, Annex E specifies syntax and semantics of the video usability information parameters of the sequence parameter set.

Throughout this specification, statements appearing with the preamble "NOTE -" are informative and are not an integral part of this Recommendation | International Standard.

1 Scope

This document specifies ITU-T Recommendation H.264 | ISO/IEC International Standard ISO/IEC 14496-10 video coding.

2 Normative references

The following Recommendations and International Standards contain provisions that, 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 Standardisation Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

– ITU-T Recommendation T.35 (2000), Procedure for the allocation of ITU-T defined codes for non-standard facilities
– ISO/IEC 11578:1996, Annex A, Universal Unique Identifier
– ISO/CIE 10527:1991, Colorimetric Observers

3 Definitions

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

3.1 **access unit**: A set of NAL units always containing a primary coded picture. In addition to the primary coded picture, an access unit may also contain one or more redundant coded pictures or other NAL units not containing slices or slice data partitions of a coded picture. The decoding of an access unit always results in a decoded picture.

访问单元：是一个含有基本编码图像的 NAL 单元集。除了基本编码图像之外，访问单元可能还含有一个或多个冗余编码图像，或者其他不含有编码图像 slice 或者 slice 数据块的 NAL 单元。一个访问单元的解码产生一个解码图像。

3.2 **AC transform coefficient**: Any transform coefficient for which the frequency index in one or both dimensions is non-zero.

AC 变换系数：一维或者二维频率索引不为零的任何一个变换系数。

3.3 **adaptive binary arithmetic decoding process**: An entropy decoding process that recovers the values of bins from a bitstream produced by an adaptive binary arithmetic encoding process.

自适应二进制算术解码过程：从自适应二进制编码过程产生的码流中恢复二进制值的熵解码过程。

3.4 **adaptive binary arithmetic encoding process**: An entropy encoding process, not normatively specified in this Recommendation | International Standard, that codes a sequence of bins and produces a bitstream that can be decoded using the adaptive binary arithmetic decoding process.

自适应二进制算术编码过程：一个熵编码过程，在本协议中没有标准的说明，该过程编码一个二进制序列并且产生一个自适应二进制算术解码过程能够解码的码流。

3.5 **arbitrary slice order**: A decoding order of slices in which the macroblock address of the first macroblock of some slice of a picture may be smaller than the macroblock address of the first macroblock of some other preceding slice of the same coded picture.

任意的 slice 顺序：一个 slice 的解码顺序，在这个解码顺序中，一些 slice 中的第一个宏块的地址也许小于相同编码图像中前面 slice 中的第一个宏块的地址。

3.6 **B slice**: A slice that may be decoded using intra prediction from decoded samples within the same slice or inter prediction from previously-decoded reference pictures, using at most two motion vectors and reference indices to predict the sample values of each block.

B slice：是一个 slice，该 slice 也许可能使用相同 slice 中的已经解码的象素进行帧内预测解码，也许使用至多两个运动矢量和参考索引根据前面解码的参考图像来预测每一个块的象素值。

3.7 **bin**: One bit of a bin string.

二进制位：一个二进制串中的一个比特位。

3.8 **binarization**: The set of intermediate binary representations of all possible values of a syntax element.

二值化：一个语法元素的所有可能值的中间二进制表示集。

3.9 **binarization process**: A unique mapping process of possible values of a syntax element onto a set of bin strings.

二值化过程：一个语法的可能值到二进制串集中的唯一映射过程。

3.10 **bin string**: A string of bins. A bit string is an intermediate binary representation of values of syntax elements.

二进制串：一个二进制位的串，一个比特串是语法元素值的中间二进制表示。

3.11 **bi-predictive slice:** See B slice**.**

双向预测 **slice**：见 B slice

3.12 **bitstream**: A sequence of bits that forms the representation of coded pictures and associated data forming one or more coded video sequence. Bitstream is a collective term used to refer either to a NAL unit stream or a byte stream.

比特流：一个组成编码图像表示和组成一个或多个编码视频序列的相关数据的比特序列。比特流是一个表示 NAL 单元流或者字节流的统称术语。

3.13 **block**: An MxN (M-column by N-row) array of samples, or an MxN array of transform coefficients.

Block：一个 MxN的象素数组（M列，N行），或者一个 MxN 变换系数数组。

3.14 **bottom field**: One of two fields that comprise a frame. Each row of a bottom field is spatially located immediately below a corresponding row of a top field.

底场：组成一帧的两场中的一场。底场中的每一行位于顶场中对应行的下边。

3.15 **bottom macroblock (of a macroblock pair)**: The macroblock within a macroblock pair that contains the samples in the bottom row of samples for the macroblock pair. For a field macroblock pair, the bottom macroblock represents the samples from the region of the bottom field of the frame that lie within the spatial region of the macroblock pair. For a frame macroblock pair, the bottom macroblock represents the samples of the frame that lie within the bottom half of the spatial region of the macroblock pair.

底场宏块（宏块对中）：在宏块对中，含有底行象素的宏块。对于一个场宏块对来说，底场宏块表示图像帧中位于宏块对位置区域的底场部分的象素。对于一个帧宏块对来说，底场宏块表示了图像帧中位于宏块对位置区域的下面一个宏块部分的象素。

3.16 **broken link**: A location in a bitstream at which it is indicated that some subsequent pictures in decoding order may contain serious visual artefacts due to unspecified operations performed in the generation of the bitstream.

中止连接：表示码流中的一个位置，按照解码顺序，从这个位置开始接下来的一些图像由于在产生码流时的不可知的操作有严重的视觉变形。（？？？？）

3.17 **byte**: A sequence of 8 bits, written and read with the most significant bit on the left and the least significant bit on the right. When represented in a sequence of data bits, the most significant bit of a byte is first.

字节：一个 8 比特的序列，从左边最重要的比特位开始向右边最不重要的比特位进行读和写操作。当出现数据比特序列时，首先出现字节中最重要的比特位。

3.18 **byte-aligned**: A bit in a bitstream is byte-aligned when its position is a multiple of 8 bits from the first bit in the bitstream.

字节对齐：在码流中从第一个比特开始每 8 个比特整数倍的位置是字节对齐的位置，那个位置的比特称为字节对齐比特。

3.19 **byte stream**: An encapsulation of a NAL unit stream containing start code prefixes and NAL units as specified in Annex B.

字节流：一个 NAL 单元的封装，包含有前缀开始码和 NAL 单元。见附录 B 所示。

3.20 **category**: A number associated with each syntax element. The category is used to specify the allocation of syntax elements to NAL units for slice data partitioning. It may also be used in a manner determined by the application to refer to classes of syntax elements in a manner not specified in this Recommendation | International Standard.

类别：和每个语法元素相关联的一个数，类别被用来说明 slice 数据分割中语法元素到 NAL 单元的分配过程。它也可以使用本协议中没有说明的语法元素分类方法，而是根据具体的应用来分类。

3.21 **chroma**: An adjective specifying that a sample array or single sample is representing one of the two colour difference signals related to the primary colours. The symbols used for a chroma array or sample are Cb and Cr.

色差：与一个基色相关的色差信号，用于对一个象素数组或者一个象素的辅助说明。表示色差数组或者色差象素的符号是：Cb 和 Cr

NOTE - The term chroma is used rather than the term chrominance in order to avoid the implication of the use of linear light transfer characteristics that is often associated with the term chrominance.

3.22 **coded field**: A coded representation of a field.

编码场：一个场的编码表示

3.23 **coded frame**: A coded representation of a frame.

编码帧：一帧图像的编码表示

3.24 **coded picture**: A coded representation of a picture. A coded picture may be either a coded field or a coded frame. Coded picture is a collective term referring to a primary coded picture or a redundant coded picture, but not to both together.

编码图像：一个图像的编码表示。一个编码图像可能是一个编码场，也可能是一个编码帧。编码图像是基本编码图像或者冗余编码图像的统称，但是同时不能表示两者。

3.25 **coded picture buffer (CPB)**: A first-in first-out buffer containing access units in decoding order specified in the hypothetical reference decoder in Annex C.

编码图像缓冲区：一个先进先出的缓冲区，该缓冲区中含有按照假设参考解码中说明的解码顺序排列的访问单元，如附录 C 所示。

3.26 **coded representation**: A data element as represented in its coded form.

编码表示：按照编码形式表示的数据元素。

3.27 **coded video sequence**: A sequence of access units that consists, in decoding order, of an IDR access unit followed zero or more non-IDR access units including all subsequent access units up to but not including any subsequent IDR access unit.

编码视频序列：一个访问单元的序列，按照解码顺序，由一个 IDR 访问单元和后面接着的零个或者多个非 IDR 访问单元组成， 它包括所有接下来的访问单元，一直到但不包括接下来的 IDR 图像。

3.28 **component**: An array or single sample from one of the three arrays (luma and two chroma) that make up a field or frame.

分量：一个象素数组或者单个象素，它是组成一帧或一场图像的三个分量中（一个亮度和两个色差）的一个。

3.29 **complementary non-reference field pair**: Two non-reference fields that are in consecutive access units in decoding order as two coded fields of opposite parity where the first field is not already a paired field.

互补的非参考场对：表示两个非参考的相对的场（如一个为顶场，另一个为底场），并且第一个场不是一个成对的场，按照解码顺序它们被包含在两个连续的访问单元中。（但这两个场不一定有相同的帧号，即不一定属于同一帧的两个场）

3.30 **complementary reference field pair**: Two reference fields that are in consecutive access units in decoding order as two coded fields and share the same value of frame number.

互补的参考场对：两个有相同帧号的参考场，按照解码顺序它们作为两个编码场被包含在连续的访问单元中。（互补的非参考场对和互补的参考场对的定义的理解？）

3.31 **context variable**: A variable specified for the adaptive binary arithmetic decoding process of a bin by an equation containing recently decoded bins.

上下文变量：一个变量，用于说明使用包含有最近解码比特位的等式进行比特位的自适应二进制算术解码过程。

3.32 **DC transform coefficient**: A transform coefficient for which the frequency index is zero in all dimensions.

DC 变换系数：所有频率索引都为零的变换系数。

3.33 **decoded picture**: A decoded picture is derived by decoding a coded picture. A decoded picture is either a decoded frame, or a decoded field. A decoded field is either a decoded top field or a decoded bottom field.

解码图像：由解码一个编码图像得到的解码图像。一个解码图像或者是一个解码帧，或者是一个解码场。一个解码场可以是解码顶场，也可以是解码底场。

3.34 **decoded picture buffer (DPB)**: A buffer holding decoded pictures for reference, output reordering, or output delay specified for the hypothetical reference decoder in Annex C.

解码图像缓冲：一个缓冲区，用于保存参考，输出重排或者输出显示的解码图像。见附录 C 中假设参考解码器。

3.35 **decoder**: An embodiment of a decoding process.

解码器：一个解码过程的具体实现。

3.36 **decoding order**: The order in which syntax elements are processed by the decoding process.

解码顺序：解码过程中语法元素处理的顺序。

3.37 **decoding process**: The process specified in this Recommendation | International Standard that reads a bitstream and produces decoded pictures.

解码过程：本协议中说明的过程，读入码流，输出解码图像。

3.38 **direct prediction**: An inter prediction for a block for which no motion vector is decoded. Two direct prediction modes are specified that are referred to as spatial direct prediction and temporal prediction mode.

直接预测：一个块的帧间预测，但是没有解码运动矢量。有两种类型的直接预测模式，分别称为空间直接预测模式和时间直接预测模式。

3.39 **decoder under test (DUT)**: A decoder that is tested for conformance to this Recommendation | International Standard by operating the hypothetical stream scheduler to deliver a conforming bitstream to the decoder and to the hypothetical reference decoder and comparing the values and timing of the output of the two decoders.

测试解码器（DUT）：用于测试和本协议兼容性的一个解码器，通过假设流调度方法给解码器和假设参考解码器提供兼容的码流，然后比较两个解码器的输出时间和解码值。

3.40 **emulation prevention byte**: A byte equal to 0x03 that may be present within a NAL unit. The presence of emulation prevention bytes ensures that no sequence of consecutive byte-aligned bytes in the NAL unit contains a start code prefix.

预防混淆字节：在 NAL 单元中可能出现的值为 0x03 的字节。预防混淆字节保证了在 NAL 单元中没有包含前缀开始码的字节对齐的连续字节序列。

3.41 **encoder**: An embodiment of an encoding process.

编码器：一个编码过程的具体实现。

3.42 **encoding process**: A process, not specified in this Recommendation | International Standard, that produces a bitstream conforming to this Recommendation | International Standard.

编码过程：产生和本协议兼容的码流的一个过程，但在本协议中没有具体说明。

3.43 **field**: An assembly of alternate rows of a frame. A frame is composed of two fields, a top field and a bottom field.

场：一帧中隔行的集合。一帧由两场组成，包括顶场和底场。

3.44 **field macroblock**: A macroblock containing samples from a single field. All macroblocks of a coded field are field macroblocks. When macroblock-adaptive frame/field decoding is in use, some macroblocks of a coded frame may be field macroblocks.

场宏块：只包含有属于同一场象素的宏块。一编码场的所有宏块都是场宏块。当使用宏块级的自适应帧场解码时，一编码帧的一些宏块可能是场宏块。

3.45 **field macroblock pair**: A macroblock pair decoded as two field macroblocks.

场宏块对：作为两个场宏块解码的宏块对。

3.46 **field scan**: A specific sequential ordering of transform coefficients that differs from the zig-zag scan by scanning columns more rapidly than rows. Field scan is used for transform coefficients in field macroblocks.

场扫描：一种变换系数的扫描顺序，和 zig-zag 扫描方式不一样，这种方式扫描列比扫描行快。对于场宏块的变换系数使用场扫描。

3.47 **flag**: A variable that can take one of the two possible values 0 and 1.

变量：有两种可能值 0 或1 的一个变量。

3.48 **frame**: A frame contains an array of luma samples and two corresponding arrays of chroma samples. A frame consists of two fields, a top field and a bottom field.

帧：含有一个亮度数组和两个对应色差数组的帧。一帧由两场组成：一个顶场和一个底场。

3.49 **frame macroblock**: A macroblock representing samples from two fields of a coded frame. When macroblock-adaptive frame/field decoding is not in use, all macroblocks of a coded frame are frame macroblocks. When macroblock-adaptive frame/field decoding is in use, some macroblocks of a coded frame may be frame macroblocks.

帧宏块：表示一编码帧中两场象素的一个宏块。当不使用宏块级的帧场自适应解码时，编码帧中的所有宏块都是帧宏块。当使用宏块级的帧场自适应解码时，那么编码帧中的一些宏块可能是帧宏块。

3.50 **frame macroblock pair**: A macroblock pair decoded as two frame macroblocks.

帧宏块对：作为两个帧宏块解码的宏块对。

3.51 **frequency index**: A one-dimensional or two-dimensional index associated with a transform coefficient prior to an inverse transform part of the decoding process.

频率索引：解码过程中，反变换之前的和一变换系数相关的一维或二维索引。

3.52 **hypothetical reference decoder (HRD)**: A hypothetical decoder model that specifies constraints on the variability of conforming NAL unit streams or conforming byte streams that an encoding process may produce.

假设参考解码（HRD）：一个假设解码模型，说明了对编码过程产生的 NAL 单元流或字节流的变化的限制。

3.53 **hypothetical stream scheduler (HSS)**: A hypothetical delivery mechanism for the timing and data flow of the input of a bitstream into the hypothetical reference decoder. The HSS is used for checking the conformance of a bitstream or a decoder.

假设流调度方法（HSS）：一种从码流输入的数据流到假设参考解码的假设传输机制。HSS 被用来检测码流或者解码器的兼容性。

3.54 **I slice**: A slice that is decoded using prediction only from decoded samples within the same slice.

I slice：解码过程中，仅仅使用来自相同 slice 中的解码象素进行预测解码的 slice。

3.55 **instantaneous decoding refresh (IDR) access unit**: An access unit in which the primary coded picture is an IDR picture.

即时解码刷新（IDR）访问单元：基本编码图像是 IDR 图像的访问单元。

3.56 **instantaneous decoding refresh (IDR) picture**: A coded picture containing only slices with I or SI slice types that causes the decoding process to mark all reference pictures as "unused for reference" immediately after decoding the IDR picture. After the decoding of an IDR picture all following coded pictures in decoding order can be decoded without inter prediction from any picture decoded prior to the IDR picture. The first picture of each coded video sequence is an IDR picture.

即时解码刷新（IDR）图像：仅仅包含 I 或 SI slice 的编码图像，解完 IDR 图像之后，解码过程立即将所有的参考图像都标识为“不用作参考”，按照解码顺序，该 IDR 图像之后的编码图像都不使用该 IDR 图像之前的任何一个解码图像作为参考。每一个编码视频序列的第一个图像是 IDR 图像。

3.57 **inter coding**: Coding of a block, macroblock, slice, or picture that uses inter prediction.

帧间编码：使用帧间预测编码一个块、宏块、slice 或 图像。

3.58 **inter prediction**: A prediction derived from decoded samples of reference pictures other than the current decoded picture.

帧间预测：使用参考图像的解码象素而不是当前图像的解码象素进行的预测。

3.59 **intra coding**: Coding of a block, macroblock, slice, or picture that uses intra prediction.

帧内编码：使用帧内预测编码一个块、宏块、slice 或者图像。

3.60 **intra prediction**: A prediction derived from the decoded samples of the same decoded slice.

帧内预测：使用来自相同 slice 的解码象素进行的预测。

3.61 **intra slice**: See I slice.

Intra slice: 见 I slice。

3.62 **inverse transform**: A part of the decoding process by which a set of transform coefficients are converted into spatial-domain values, or by which a set of transform coefficients are converted into DC transform coefficients.

反变换：是解码过程的一部分，将变换系数集变换成空域值，或者将变换系数值变换成 DC 变换系数。

3.63 **layer**: One of a set of syntactical structures in a non-branching hierarchical relationship. Higher layers contain lower layers. The coding layers are the coded video sequence, picture, slice, and macroblock layers.

层：是非分枝层状关系中的一个语法结构集，高层包含低层。编码层是编码视频序列、图像、slice 和宏块层。

3.64 **level**: A defined set of constraints on the values that may be taken by the syntax elements and variables of this Recommendation | International Standard. The same set of levels is defined for all profiles, with most aspects of the definition of each level being in common across different profiles. Individual implementations may, within specified constraints, support a different level for each supported profile. In a different context, level is the value of a transform coefficient prior to scaling.

级：对本协议中的语法元素和变量定义的限制集。相同的集是面向所有框架进行定义的，每一个集的大部分在不同框架之间都是共有的。在说明的限制中，对于每个支持的框架，个别实现也许支持不同的级。在不同的文中，级也是缩放之前变换系数的值。

3.65 **list 0 (list 1) motion vector**: A motion vector associated with a reference index pointing into reference picture list 0 (list 1).

列表 0 （列表 1）运动矢量：和指向参考图像列表 0 （列表1）的一参考图像索引相关的运动矢量。

3.66 **list 0 (list 1) prediction**: Inter prediction of the content of a slice using a reference index pointing into reference picture list 0 (list 1).

列表 0 （列表 1）预测：用指向参考图像列表 0（列表 1）的参考索引进行 slice 内容的帧间预测。

3.67 **luma**: An adjective specifying that a sample array or single sample is representing the monochrome signal related to the primary colours. The symbol used for luma is Y.

亮度：用来表示和基本色相关的单色信号的单个象素或象素数组。表示亮度的符号是：Y。

NOTE – The term luma is used rather than the term luminance in order to avoid the implication of the use of linear light transfer characteristics that is often associated with the term luminance.

3.68 **macroblock**: A 16x16 block of luma samples and two corresponding blocks of chroma samples. The division of a slice or a macroblock pair into macroblocks is a partitioning.

宏块：一个 16X16 的亮度象素块和两个对应的色差象素块。一个 slice 或者一个宏块对到宏块的分割称为划分。

3.69 **macroblock-adaptive frame/field decoding**: A decoding process for coded frames in which some macroblocks may be decoded as frame macroblocks and others may be decoded as field macroblocks.

宏块自适应帧场解码：指一编码帧的解码过程，其中一些宏块解码成帧宏块，另外一些宏块解码成场宏块。

3.70 **macroblock address**: When macroblock-adaptive frame/field decoding is not in use, a macroblock address is the index of a macroblock in a macroblock raster scan of the picture starting with zero for the top-left macroblock in a picture. When macroblock-adaptive frame/field decoding is in use, the macroblock address of the top macroblock of a macroblock pair is two times the index of the macroblock pair in a macroblock pair raster scan of the picture, and the macroblock address of the bottom macroblock of a macroblock pair is the macroblock address of the corresponding top macroblock plus 1. The macroblock address of the top macroblock of each macroblock pair is an even number and the macroblock address of the bottom macroblock of each macroblock pair is an odd number.

宏块地址：当不使用宏块自适应帧场解码时，宏块地址是图像中按照宏块栅格扫描的宏块索引，从图像中地址为 0 左上角的宏块开始扫描。当使用宏块自适应帧场解码时，宏块对的顶场宏块的宏块地址是图像中按照宏块对栅格扫描的宏块对索引的两倍。宏块对的底场宏块的宏块地址是对应的顶场宏块地址加 1，每个宏块对顶场宏块的宏块地址是偶数而底场宏块的宏块地址是奇数。

3.71 **macroblock location**: The two-dimensional coordinates of a macroblock in a picture denoted by (x, y). For the top left macroblock of the picture (x, y) is equal to (0, 0). x is incremented by 1 for each macroblock column from left to right. When macroblock-adaptive frame/field decoding is not in use, y is incremented by 1 for each macroblock row from top to bottom. When macroblock-adaptive frame/field decoding is in use, y is incremented by 2 for each macroblock pair row from top to bottom, and is incremented by an additional 1 when a macroblock is a bottom macroblock.

宏块位置：宏块在图像中的二维坐标，表示为(x, y)。对于图像中左上角的宏块，(x, y)等于(0, 0)。对于从左到右的每一个宏块，x依次加 1。当不使用宏块自适应帧场解码时，对于从上到下的每一个宏块，y 依次加 1，当使用宏块自适应帧场解码时，对于从上到下的每一个宏块对，y 依次加 2，当宏块为底场宏块时，y 再被加 1。

3.72 **macroblock pair**: A pair of vertically contiguous macroblocks in a frame that is coupled for use in macroblock-adaptive frame/field decoding processing. The division of a slice into macroblock pairs is a partitioning.

宏块对：指的是在使用宏块自适应帧场解码时在一帧中垂直连续的宏块对。Slice 到宏块对的分割也被称为划分。

3.73 **macroblock partition**: A block of luma samples and two corresponding blocks of chroma samples resulting from a partitioning of a macroblock for inter prediction.

宏块部分：在宏块的帧间预测中对宏块的划分而得到的一个亮度象素块和两个对应的色差象素块。

3.74 **macroblock to slice group map**: A means of mapping macroblocks of a picture into slice groups. The macroblock to slice group map consists of a list of numbers, one for each coded macroblock, specifying the slice group to which each coded macroblock belongs.

宏块到 slice 组映射：一种将图像中的宏块映射到 slice 组的方法。宏块到 slice 组的映射由一列数组成，每个数对应一个编码宏块，表示这个宏块属于哪个 slice 组。

3.75 **map unit to slice group map**: A means of mapping slice group map units of a picture into slice groups. The map unit to slice group map consists of a list of numbers, one for each slice group map unit, specifying the slice group to which each coded slice group map unit belongs.

映射单元到 slice 组的映射：一种将图像中slice 组映射单元映射到 slice 组的方法。映射单元到 slice 组的映射由一列数组成，每个数对应一个 slice 组映射单元，表示这个 slice 组映射单元属于哪个 slice 组。

3.76 **memory management control operation**: Seven operations that control reference picture marking.

缓存管理控制操作：控制参考图像标识的七种操作。

3.77 **motion vector**: A two-dimensional vector used for inter prediction that provides an offset from the coordinates in the decoded picture to the coordinates in a reference picture.

运动矢量：用作帧间预测的二维矢量，表示了解码图像中的坐标到参考图像中的坐标的偏移。

3.78 **NAL unit**: A syntax structure containing an indication of the type of data to follow and bytes containing that data in the form of an RBSP interspersed as necessary with emulation prevention bytes.

NAL 单元：一种语法结构，包含有数据类型的说明和使用了防止混淆字节的RBSP 形式的数据。

3.79 **NAL unit stream**: A sequence of NAL units.

NAL 单元流：一个 NAL 单元序列。

3.80 **non-paired reference field**: A decoded reference field that is not part of a complementary reference field pair.

非成对的参考场：指一解码参考场，但它不是一互补参考场的一部分。

3.81 **non-reference picture**: A picture coded with nal_ref_idc equal to 0. A non-reference picture is not used for inter prediction of any other pictures.

非参考图像：nal_ref_idc 等于 0 的图像。非参考图像不用作其它任何图像的帧间预测。

3.82 **opposite parity**: The opposite parity of top is bottom, and vice versa.

相反奇偶性：顶场的相反奇偶性是底场，反之也成立。

3.83 **output order**: The order in which the decoded pictures are output from the decoded picture buffer.

输出顺序：从解码缓冲区中输出解码图像的顺序。

3.84 **P slice**: A slice that may be decoded using intra prediction from decoded samples within the same slice or inter prediction from previously-decoded reference pictures, using at most one motion vector and reference index to predict the sample values of each block.

P slice：一个 slice，可能使用相同 slice 中的解码象素进行帧内预测解码或者使用前面解码的参考图像进行帧间预测解码，但最多只使用一个运动矢量和参考索引来预测每一个块的象素。

3.85 **parameter**: A syntax element of a sequence parameter set or a picture parameter set. Parameter is also used as part of the defined term quantisation parameter.

参数：序列参数集或者图像参数集中的语法元素。参数也被用作定义的量化参数的一部分。

3.86 **parity**: The parity of a field can be top or bottom.

奇偶：场的奇偶可能是顶场或底场。

3.87 **partitioning**: The division of a set into subsets such that each element of the set is in exactly one of the subsets.

划分：一个集被分割成一个子集，以至于集中的元素一定在一个子集中。

3.88 **picture**: A collective term for a field or a frame.

图像：帧和场的统称。

3.89 **picture order count**: A variable having a value that increases with increasing picture position in output order relative to the previous IDR picture in decoding order or relative to the previous picture containing the memory management control operation that marks all reference pictures as “unused for reference”.

图像顺序数：表示在输出顺序中图像位置的变量，随着相对于解码顺序上的前一个 IDR图像或者缓冲区管理控制操作为标识所有的参考图像为“不用作参考“的前一图像的图像位置的增加，这个值也随着增加。

3.90 **prediction**: An embodiment of the prediction process.

预测：预测过程的具体实现。

3.91 **prediction process**: The use of a predictor to provide an estimate of the sample value or data element currently being decoded.

预测过程：预测器的使用，提供当前解码的数据元素或者象素值的估计。

3.92 **predictive slice**: See P slice.

预测 slice：见 P slice。

3.93 **predictor**: A combination of previously decoded sample values or data elements used in the decoding process of subsequent sample values or data elements.

预测器：在解码接下来的象素值和数据元素过程中使用的前面已经解码的象素值或数据元素的组合。

3.94 **primary coded picture**: The coded representation of a picture to be used by the decoding process for a bitstream conforming to this Recommendation | International Standard. The primary coded picture contains all macroblocks of the picture. The only pictures that have a normative effect on the decoding process are primary coded pictures. See also redundant coded picture.

基本编码图像：图像的编码表示，可以被遵循本协议的解码过程解码。基本编码图像包含了图像中的所有宏块。只有对解码过程有规范效果的图像才是基本编码图像。见冗余编码图像。

3.95 **profile**: A specified subset of the syntax of this Recommendation | International Standard.

框架：本协议的语法子集的说明。

3.96 **quantisation parameter**: A variable used by the decoding process for scaling of transform coefficient levels.

量化参数：对于变换系数值缩放的解码过程中使用的变量。

3.97 **random access**: The act of starting the decoding process for a bitstream at a point other than the beginning of the stream.

随机访问：不从码流的开始位置处进行码流的解码。

3.98 **raster scan**: A mapping of a rectangular two-dimensional pattern to a one-dimensional pattern such that the first entries in the one-dimensional pattern are from the first top row of the two-dimensional pattern scanned from left to right, followed similarly by the second, third, etc. rows of the pattern (going down) each scanned from left to right.

栅格扫描：一个从矩形二维模式到一维模式的映射，一维模式中的第一个索引是二维模式中的最上边一行最左边的元素，然后从左到右，从上到下依次扫描成一维模式中的第二个索引、第三个索引等等。

3.99 **raw byte sequence payload (RBSP):** A syntax structure containing an integer number of bytes that is encapsulated in a NAL unit. An RBSP is either empty or has the form of a string of data bits containing syntax elements followed by an RBSP stop bit and followed by zero or more subsequent bits equal to 0.

原始字节序列负载（RBSP）：包含有封装在 NAL 单元中的整数个字节的语法结构。一个 RBSP 可以为空，或者为包含有语法元素的数据比特串，后面紧跟着 RBSP 停止比特位和零个或者多个等于零的比特位。

3.100 **raw byte sequence payload (RBSP) stop bit:** A bit equal to 1 present within a raw byte sequence payload (RBSP) after a string of data bits. The location of the end of the string of data bits within an RBSP can be identified by searching from the end of the RBSP for the RBSP stop bit, which is the last non-zero bit in the RBSP.

原始字节序列负载（RBSP）停止比特位：在原始字节序列负载中数据比特串之后的等于 1 的比特位。通过搜索RBSP结尾处的RBSP停止比特位，可以识别出在RBSP中数据比特串结尾的位置，它是RBSP中的最后一个非零比特位。

3.101 **recovery point**: A point in the bitstream at which the recovery of an exact or an approximate representation of the decoded pictures represented by the bitstream is achieved after a random access or broken link.

恢复点：表示比特流中的一个位置，随机访问或者中止连接之后，从这个位置可以准确或者近似地恢复出码流表示的解码图像。

3.102 **redundant coded picture**: A coded representation of a picture or a part of a picture. The content of a redundant coded picture shall not be used by the decoding process for a bitstream conforming to this Recommendation | International Standard. A redundant coded picture is not required to contain all macroblocks in the primary coded picture. Redundant coded pictures have no normative effect on the decoding process. See also primary coded picture.

冗余编码图像：一图像或者部分图像的编码表示。对于遵循本协议的码流来说，解码过程可以不使用冗余编码图像的内容。冗余编码图像不一定包含有基本编码图像中的所有宏块。冗余编码图像对解码过程没有规定的效果。见基本编码图像。

3.103 **reference field**: A reference field may be used for inter prediction when P, SP, and B slices of a coded field or field macroblocks of a coded frame are decoded. See also reference picture.

参考场：当解码编码场的 P、SP 和 B slice 或编码帧的场宏块时，可能使用参考场来做帧间预测。见参考图像。

3.104 **reference frame**: A reference frame may be used for inter prediction when P, SP, and B slices of a coded frame are decoded. See also reference picture.

参考帧：当解码编码帧的 P、SP 和 B slice 时可能使用参考帧来做帧间预测，见参考图像。

3.105 **reference index**: An index into a reference picture list.

参考索引：参考图像列表中的一个索引。

3.106 **reference picture**: A picture with nal_ref_idc not equal to 0. A reference picture contains samples that may be used for inter prediction in the decoding process of subsequent pictures in decoding order.

参考图像：nal_ref_idc 不等于 0 的图像。解码接下来的图像时，解码过程可能使用参考图像来做帧间预测。

3.107 **reference picture list**: A list of short-term picture numbers and long-term picture numbers that are assigned to reference pictures.

参考图像列表：短期图像数和长期图像数的一个列表，这个列表被分配给参考图像。

3.108 **reference picture list 0**: A reference picture list used for inter prediction of a P, B, or SP slice. All inter prediction used for P and SP slices uses reference picture list 0. Reference picture list 0 is one of two reference picture lists used for inter prediction for a B slice, with the other being reference picture list 1.

参考图像列表 0：用作 P, B, 或者 SP slice 帧间预测的参考图像列表。P 和 SP slice 的所有帧间预测都使用参考图像列表 0。在 B slice 的帧间预测中，参考图像列表 0 是两个参考图像列表中的一个，另外一个是参考图像列表 1。

3.109 **reference picture list 1**: A reference picture list used for inter prediction of a B slice. Reference picture list 1 is one of two lists of reference picture lists used for inter prediction for a B slice, with the other being reference picture list 0.

参考图像列表 1：在 B slice 的帧间预测中，使用参考图像列表 1 来做帧间预测，参考图像列表 1 是两个参考图像列表中的一个，另外一个是参考图像列表 0。

3.110 **reference picture marking**: Specifies, in the bitstream, how the decoded pictures are marked for inter prediction.

参考图像标识：在码流中说明用作帧间预测的解码图像是如何标识的。

3.111 **reserved**: The term “reserved”, when used in the clauses specifying some values of a particular syntax element, means that these values shall not be used in bitstreams conforming to this Recommendation | International Standard, but may be used in future extensions of this Recommendation | International Standard by ITU-T | ISO/IEC.

保留：在说明一些具体语法元素的值时，“保留”意味着这个值在遵循本协议的码流中还没有被使用，但是可能在本协议的扩展版本中使用。

3.112 **residual**: The decoded difference between a prediction of a sample or data element and its decoded value.

残差：象素或者数据元素的预测值和它的解码值之间的差。

3.113 **run**: A number of consecutive data elements represented in the decoding process. In one context, the number of zero-valued transform coefficient levels preceding a non-zero transform coefficient level in the list of transform coefficient levels generated by a zig-zag scan or a field scan. In other contexts, run refers to a number of macroblocks.

行程：在解码过程中出现的连续数据元素的数目。可能表示的是由 zig-zag 扫描或场扫描产生的变换系数列表中的非零变换系数之前零值变换系数的数目，也可能表示的是宏块的数目。

3.114 **sample aspect ratio**: Specifies, for assisting the display process, which is not specified in this Recommendation | International Standard, the ratio between the intended horizontal distance between the columns and the intended vertical distance between the rows of the luma sample array in a frame. Sample aspect ratio is expressed as h:v, where h is horizontal width and v is vertical height (in arbitrary units of spatial distance).

采样宽高比：为了协助本协议中没有说明的显示过程，采样宽高比表示了图像帧中亮度象素数组列间的水平距离和行间的垂直距离之间的比例关系。采样宽高比被表示成 h:v，这里 h 是水平宽度，v 是垂直高度（单位是任意的空间距离）。

3.115 **scaling**: The process of multiplying transform coefficient levels by a factor, resulting in transform coefficients.

缩放：表示了变换系数值和一个系数相乘的过程，得到变换系数。

3.116 **SI slice**: A slice that is coded using prediction only from decoded samples within the same slice and using quantisation of the prediction samples. An SI slice can be coded such that its decoded samples can be constructed identically to an SP slice.

SI slice：表示仅仅使用相同 slice 中的解码象素和预测象素的量化值来预测编码的 slice。SI slice 编码保证了它的解码象素值和 SP slice 的解码象素值相同。

3.117 **skipped macroblock**: A macroblock for which no data is coded other than an indication that the macroblock is to be decoded as "skipped". This indication may be common to several macroblocks.

跳跃宏块：表示除了标识为"跳跃宏块"的标志位以外，没有编码其它任何数据的宏块。这个标志位也许是几个宏块共有的。

3.118 **slice**: An integer number of macroblocks or macroblock pairs ordered consecutively in the raster scan within a particular slice group. For the primary coded picture, the division of each slice group into slices is a partitioning. Although a slice contains macroblocks or macroblock pairs that are consecutive in the raster scan within a slice group, these macroblocks or macroblock pairs are not necessarily consecutive in the raster scan within the picture. The addresses of the macroblocks are derived from the address of the first macroblock in a slice (as represented in the slice header) and the macroblock to slice group map.

Slice：在具体的 slice 组中，按照栅格扫描的整数个连续的宏块或者宏块对。对于基本编码图像来说，slice 组到 slice 的分割称为划分，虽然每个 slice 包含有 slice 组中按照栅格扫描顺序连续的宏块或宏块对，但是这些宏块或宏块对在图像中按照栅格扫描顺序没有必要连续。宏块地址取决于 slice 中的第一个宏块的地址（在 slice 中表示）和宏块到 slice 组的映射。

3.119 **slice data partitioning**: A method of partitioning selected syntax elements into syntax structures based on a category associated with each syntax element.

Slice 数据分割：根据和每个语法元素相关的类别，将选择的语法元素划分到对应的语法结构中的方法。

3.120 **slice group**: A subset of the macroblocks or macroblock pairs of a picture. The division of the picture into slice groups is a partitioning of the picture. The partitioning is specified by the macroblock to slice group map.

Slice 组：图像中宏块或者宏块对的子集。将图像分割成 slice 组是图像的划分，这个划分由宏块到 slice 组映射来说明。

3.121 **slice group map units**: The units of the map unit to slice group map.

Slice 组映射单元：映射到 slice 组的单元。

3.122 **slice header**: A part of a coded slice containing the data elements pertaining to the first or all macroblocks represented in the slice.

slice 头：表示编码 slice 的一部分，包含有属于 slice 中第一个宏块或者所有宏块的数据元素。

3.123 **source**: Term used to describe the video material or some of its attributes before encoding.

信源：用于描述视频源或者编码之前视频源的一些属性。

3.124 **SP slice**: A slice that is coded using inter prediction from previously-decoded reference pictures, using at most one motion vector and reference index to predict the sample values of each block. An SP slice can be coded such that its decoded samples can be constructed identically to another SP slice or an SI slice.

SP slice：使用前面解码的参考图像进行帧间预测编码的 slice，最多只使用一个运动矢量和一个参考索引来预测每一个块的象素。SP slice 的编码保证了它的解码象素和其它的 SP slice 或者 SI slice 相同。

3.125 **start code prefix**: A unique sequence of three bytes equal to 0x000001 embedded in the byte stream as a prefix to each NAL unit. The location of a start code prefix can be used by a decoder to identify the beginning of a new NAL unit and the end of a previous NAL unit. Emulation of start code prefixes is prevented within NAL units by the inclusion of emulation prevention bytes.

开始码前缀：一个唯一的等于 0x000001 的三字节序列，嵌入在字节流中，作为每一个 NAL 单元的前缀。解码过程可以使用开始码前缀的位置来识别一个新NAL 单元的开始和前一个 NAL 单元的结束。在 NAL 单元中通过插入预防混淆字节可以防止前缀码的混淆。

3.126 **string of data bits (SODB)**: A sequence of some number of bits representing syntax elements present within a raw byte sequence payload prior to the raw byte sequence payload stop bit. Within an SODB, the left-most bit is considered to be the first and most significant bit, and the right-most bit is considered to be the last and least significant bit.

数据比特串（SODB）：在原始字节序列负载中，原始字节序列负载停止位之前的表示语法元素的一定数目的比特串。在 SODB 中，最左边的比特位是第一个也是最重要的比特位，最右边的比特位是最后一个也是最不重要的比特位。

3.127 **sub-macroblock**: One quarter of the samples of a macroblock, i.e., an 8x8 luma block and two 4x4 chroma blocks of which one corner is located at a corner of the macroblock.

子宏块：一个宏块的 1/4 象素，例如：一个 8x8 亮度块和两个 4x4 色差块，它们的一个顶点是宏块的一个顶点。

3.128 **sub-macroblock partition**: A block of luma samples and two corresponding blocks of chroma samples resulting from a partitioning of a sub-macroblock for inter prediction.

子宏块划分：表示在帧间预测的过程中对子宏块的划分导致的一个亮度块和两个对应的色差块。

3.129 **switching I slice**: See SI slice.

切换帧内 slice：见 SI slice。

3.130 **switching P slice**: See SP slice.

切换帧间 slice：见 SP slice。

3.131 **syntax element**: An element of data represented in the bitstream.

语法元素：码流中数据元素。

3.132 **syntax structure**: Zero or more syntax elements present together in the bitstream in a specified order.

语法结构：按照说明的顺序，一起出现在码流中的零个或多个语法元素。

3.133 **top field**: One of two fields that comprise a frame. Each row of a top field is spatially located immediately above the corresponding row of the bottom field.

顶场：组成帧的两场中的一场，顶场中的每一行位于底场中对应行的上面并和它相邻。

3.134 **top macroblock (of a macroblock pair)**: The macroblock within a macroblock pair that contains the samples in the top row of samples for the macroblock pair. For a field macroblock pair, the top macroblock represents the samples from the region of the top field of the frame that lie within the spatial region of the macroblock pair. For a frame macroblock pair, the top macroblock represents the samples of the frame that lie within the top half of the spatial region of the macroblock pair.

顶场宏块（宏块对）：宏块对中的一个宏块，还有宏块对中顶行的象素。对于一个场宏块对来说，顶场宏块表示图像帧中位于宏块对位置区域的顶场部分的象素。对于一个帧宏块对来说，顶场宏块表示了图像帧中位于宏块对位置区域的上面一个宏块部分的象素。

3.135 **transform coefficient**: A scalar quantity, considered to be in a frequency domain, that is associated with a particular one-dimensional or two-dimensional frequency index in an inverse transform part of the decoding process.

变换系数：频率域中值，在解码过程的反变换中和具体的一维或二维频率索引相关。

3.136 **transform coefficient level**: An integer quantity representing the value associated with a particular two-dimensional frequency index in the decoding process prior to scaling for computation of a transform coefficient value.

变换系数值：表示解码过程中缩放之前的和具体的一维或二维频率索引相关的一个整数值，用于计算变换系数值。

3.137 **universal unique identifier (UUID)**: An identifier that is unique with respect to the space of all universal unique identifiers.

通用的唯一标识符（UUID）：相对于所有的通用唯一标识符来说它是一个唯一的标识符。

3.138 **variable length coding (VLC)**: A reversible procedure for entropy coding that assigns shorter bit strings to symbols expected to be more frequent and longer bit strings to symbols expected to be less frequent.

变长编码：对于熵编码来说，是一个可逆的过程，熵编码将短比特串分配给出现频率高的字符，长比特串分配给出现频率低的字符。

3.139 **zig-zag scan**: A specific sequential ordering of transform coefficient levels from (approximately) the lowest spatial frequency to the highest. Zig-zag scan is used for transform coefficient levels in frame macroblocks.

zig-zag 扫描：变换系数值的一个特殊的排列方法，从最低的（或近似最低）空域频率到最高的空域频率进行排序。在帧宏块中对变换系数值使用 Zig-zag 扫描。

4 Abbreviations

4.1 **CABAC**: Context-based Adaptive Binary Arithmetic Coding

基于上下文自适应的二进制算术编码

4.2 **CAVLC**: Context-based Adaptive Variable Length Coding

基于上下文自适应变长编码

4.3 **CBR**: Constant Bit Rate

不变比特率

4.4 **CPB**: Coded Picture Buffer

编码图像缓冲区

4.5 **DPB**: Decoded Picture Buffer

解码图像缓冲区

4.6 **DUT**: Decoder under test

测试解码器

4.7 **FIFO**: First-In, First-Out

先进先出

4.8 **HRD**: Hypothetical Reference Decoder

假设参考解码

4.9 **HSS**: Hypothetical Stream Scheduler

假设流方法

4.10 **IDR**: Instantaneous Decoding Refresh

即时解码刷新

4.11 **LSB**: Least Significant Bit

最低有效位

4.12 **MB**: Macroblock

宏块

4.13 **MBAFF**: Macroblock-Adaptive Frame-Field Coding

宏块自适应帧场编码

4.14 **MSB**: Most Significant Bit

最高有效位

4.15 **NAL**: Network Abstraction Layer

网络抽象层

4.16 **RBSP**: Raw Byte Sequence Payload

原始比特序列负载

4.17 **SEI**: Supplemental Enhancement Information

补充增强层信息

4.18 **SODB**: String Of Data Bits

数据比特串

4.19 **UUID**: Universal Unique Identifier

通用的唯一标识

4.20 **VBR**: Variable Bit Rate

可变码率

4.21 **VCL**: Video Coding Layer

视频编码层

4.22 **VLC**: Variable Length Coding

变长编码

4.23 **VUI**: Video Usability Information

视频可用信息

5 Conventions

5 约定

NOTE - The mathematical operators used in this Specification are similar to those used in the C programming language. However, integer division and arithmetic shift operations are specifically defined. Numbering and counting conventions generally begin from 0.

注 – 在这个规范中使用的算术操作符类似与 C 编程语言中的操作符，但是这里单独定义了整除操作和算术移位操作。编号和计数习惯于从零开始。

5.1 Arithmetic operators

5.1 算术操作

The following arithmetic operators are defined as follows.

算术操作的定义如下所示：

+ Addition

+ 加法

– Subtraction (as a two-argument operator) or negation (as a unary prefix operator)

– 减法（作为双参数的操作）或者 非操作（一元的前缀操作）

* Multiplication

* 乘法

x^y Exponentiation. Specifies x to the power of y. In other contexts, such notation is used for superscripting not intended for interpretation as exponentiation.

x^y 取幂，表示了 X 的 Y 次幂。根据上下文，该表示也许用作上标，而不是作为幂次运算。

/ Integer division with truncation of the result toward zero. For example, 7/4 and –7/–4 are truncated to 1 and –7/4 and 7/–4 are truncated to –1.

/ 向零方向对结果进行截断的整除操作。例如：7/4 和 –7/–4 被截断成 1 ，–7/4 和 7/–4 被截断成 –1。

÷ Used to denote division in mathematical equations where no truncation or rounding is intended.

÷ 用来表示算术等式中的除法操作，但是这里没有截断或者四舍五入操作。

$\frac{x}{y}$ Used to denote division in mathematical equations where no truncation or rounding is intended.

$\frac{x}{y}$ 用来表示算术等式中的除法操作，但是这里没有截断或者四舍五入操作。

$\sum_{i=x}^{y} f(i)$ The summation of f(i) with i taking all integer values from x up to and including y.

用于计算 f(i) 的和，i 取 x 到 y 之间的所有整数，包括 x 和 y。

x % y Modulus. Remainder of x divided by y, defined only for integers x and y with x >= 0 and y > 0.

取模运算，x 除 y 的余数，这里 x 和 y 都必须是整数，并且 x >= 0 和 y > 0。

When order of precedence is not indicated explicitly by use of parenthesis, the following rules apply

当没有使用括号来显式的表示优先顺序，那么使用下面的规则：

– multiplication and division operations are considered to take place before addition and subtraction
– multiplication and division operations in sequence are evaluated sequentially from left to right
– addition and subtraction operations in sequence are evaluated sequentially from left to right
– 乘和除操作优先与加和减操作；
– 在乘和除操作序列中，依次从左到右进行计算；
– 在加和减操作序列中，依次从左到右进行计算；

5.2 Logical operators

5.2 逻辑操作

The following logical operators are defined as follows

逻辑操作定义如下：

x && y Boolean logical "and" of x and y

x 和 y 的布尔逻辑“与”操作

x || y Boolean logical "or" of x and y

x 和 y 的布尔逻辑“或”操作

! Boolean logical "not"

布尔逻辑“非”操作

x ? y : z If x is TRUE or not equal to 0, evaluates to the value of y; otherwise, evaluates to the value of z

如果 x 为 TRUE 或者不等于 0，那么返回 y 的值，否则，返回 z 的值。

5.3 Relational operators

5.3 关系操作

The following relational operators are defined as follows

关系操作定义如下：

> Greater than

大于

>= Greater than or equal to

大于或者等于

< Less than

小于

<= Less than or equal to

小于或者等于

= = Equal to

等于

!= Not equal to

不等于

5.4 Bit-wise operators

位逻辑操作

The following bit-wise operators are defined as follows

下面的逻辑位操作定义如下：

& Bit-wise "and". When operating on integer arguments, operates on a two's complement representation of the integer value.

逻辑“与”。当用作整数变量时，操作的是整数值的二的补码表示。

| Bit-wise "or". When operating on integer arguments, operates on a two's complement representation of the integer value.

逻辑“或”。当用作整数变量时，操作的是整数值的二的补码表示。

x >> y Arithmetic right shift of a two's complement integer representation of x by y binary digits. This function is defined only for positive values of y. Bits shifted into the MSBs as a result of the right shift shall have a value equal to the MSB of x prior to the shift operation.

X 的二的补码整数表示向右算术移动 y 个二进制位。只有 y 为正数时才有这个函数定义。右移的结果是移进 MSB 的比特位应该等于移位操作之前的 x 的 MSB。

x << y Arithmetic left shift of a two's complement integer representation of x by y binary digits. This function is defined only for positive values of y. Bits shifted into the LSBs as a result of the left shift have a value equal to 0.

X 的二的补码整数表示向左算术移动 y 个二进制位。只有 y 为正数时才有这个函数定义。左移的结果是移进 LSB 的比特位应该等于移位操作之前的 x 的 LSB。

5.5 Assignment operators

赋值操作

The following arithmetic operators are defined as follows

下面的算术操作定义如下：

= Assignment operator.

赋值操作

++ Increment, i.e., x++ is equivalent to x = x + 1; when used in an array index, evaluates to the value of the variable prior to the increment operation.

增，例如，x++ 等于 x = x + 1；当使用在数组索引中时，应该等于增操作之前变量的值。

–– Decrement, i.e., x–– is equivalent to x = x – 1; when used in an array index, evaluates to the value of the variable prior to the decrement operation.

减，例如，x–– 等于 x = x – 1；当使用在数组索引中时，应该等于减操作之前变量的值。

+= Increment by amount specified, i.e., x += 3 is equivalent to x = x + 3, and x += (-3) is equivalent to x = x + (-3).

增加说明的量，例如：x += 3 等于 x = x + 3，x += (-3) 等于 x = x + (-3)。

–= Decrement by amount specified, i.e., x –= 3 is equivalent to x = x – 3, and x –= (-3) is equivalent to x = x – (-3).

减少说明的量，例如：x –= 3 等于 x = x – 3, x –= (-3) 等于 x = x – (-3)。

5.6 Range notation

范围标注

The following notation is used to specify a range of values

下面的符号用来说明一个值的范围

x = y .. z   x takes on integer values starting from y to z inclusive, with x, y, and z being integer numbers.

x = y .. z   x 取得整数值在 y 到 z 的范围内，包括 y 和 z，这里 x，y 和 z 都是整数。

5.7 Mathematical functions

数学函数

The following mathematical functions are defined as follows

下面的数学函数定义如下：

$$\mathrm{Abs}(x) = \begin{cases} x & ; \quad x >= 0 \\ -x & ; \quad x < 0 \end{cases} \qquad (5\text{-}1)$$

Ceil(x)   the smallest integer greater than or equal to x. (5-2)

Ceil(x)   表示大于或者等于 x 的最小整数值。

$$\mathrm{Clip1}(x) = \mathrm{Clip3}(0, 255, x) \qquad (5\text{-}3)$$

$$\mathrm{Clip3}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{otherwise} \end{cases} \qquad (5\text{-}4)$$

Floor(x) the greatest integer less than or equal to x. (5-5)

Floor(x) 表示小于或者等于 x 的最大整数值

$$\mathrm{InverseRasterScan}(a, b, c, d, e) = \begin{cases} (a\%(d/b))*b; & e == 0 \\ (a/(d/b))*c; & e == 1 \end{cases} \qquad (5\text{-}6)$$

Log2(x) returns the base-2 logarithm of x. (5-7)

Log2(x) 返回 x 的以 2 为底的对数。

Log10(x) returns the base-10 logarithm of x. (5-8)

Log10(x) 返回 x 的以 10 为底的对数。

$$\mathrm{Luma4x4BlkScan}(x, y) = (x / 2) * 4 + (y / 2) * 8 + \mathrm{RasterScan}(x \% 2, y \% 2, 2) \qquad (5\text{-}9)$$

$$\mathrm{Median}(x, y, z) = x + y + z - \mathrm{Min}(x, \mathrm{Min}(y, z)) - \mathrm{Max}(x, \mathrm{Max}(y, z)) \qquad (5\text{-}10)$$

$$\mathrm{Min}(x, y) = \begin{cases} x & ; \quad x <= y \\ y & ; \quad x > y \end{cases} \qquad (5\text{-}11)$$

$$\mathrm{Max}(x, y) = \begin{cases} x & ; \quad x >= y \\ y & ; \quad x < y \end{cases} \qquad (5\text{-}12)$$

$$\text{RasterScan}(x, y, n_x) = x + y * n_x \quad (5\text{-}13)$$

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5) \quad (5\text{-}14)$$

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x >= 0 \\ -1 & ; \quad x < 0 \end{cases} \quad (5\text{-}15)$$

$$\text{Sqrt}(x) = \sqrt{x} \quad (5\text{-}16)$$

5.8 Variables, syntax elements, and tables

变量，语法元素和表

Syntax elements in the bitstream are represented in **bold** type. Each syntax element is described by its name (all lower case letters with underscore characters), its one or two syntax categories, and one or two descriptors for its method of coded representation. The decoding process behaves according to the value of the syntax element and to the values of previously decoded syntax elements. When a value of a syntax element is used in the syntax tables or the text, it appears in regular (i.e., not bold) type.

码流中的语法元素用粗体来表示，每个语法元素用它的名字、一个或两个语法类别和一个或两个编码方法来描述。解码过程根据语法元素的值和前面已经解码的语法元素的值进行解码。当在语法表中或文中使用语法元素的值时，它按照正常的字体出现（不是粗体）。

In some cases the syntax tables may use the values of other variables derived from syntax elements values. Such variables appear in the syntax tables, or text, named by a mixture of lower case and upper case letter and without any underscore characters. Variables starting with an upper case letter are derived for the decoding of the current syntax structure and all depending syntax structures. Variables starting with an upper case letter may be used in the decoding process for later syntax structures mentioning the originating syntax structure of the variable. Variables starting with a lower case letter are only used within the subclause in which they are derived.

在某些情况下，语法表中可能使用根据语法元素的值计算得到的其他变量的值。在语法表中或文中，这些变量使用混合大写和小写字母来命名，并且没有下划线。为了解码当前的语法结构和所有相关的语法结构，需要计算以大写字母开头的变量。在涉及到变量的原来语法结构的后面语法结构的解码过程中可能使用这些大写字母开头的变量。小写字母开头的变量仅仅在计算它们的那个小节中使用。

In some cases, "mnemonic" names for syntax element values or variable values are used interchangeably with their numerical values. Sometimes "mnemonic" names are used without any associated numerical values. The association of values and names is specified in the text. The names are constructed from one or more groups of letters separated by an underscore character. Each group starts with an upper case letter and may contain more upper case letters.

在某些情况下，语法元素值或变量值的名称和它们的实际值交替使用。有时，它们的名称又不和任何相关的数值使用。数值和它们名称的关系在文中说明。名称由一个或多个小写字母组合和下划线组合而成。每个字母组合都是由一个大写字母开始，并且也许包含多个大写字母。

NOTE - The syntax is described in a manner that closely follows the C-language syntactic constructs.

注－ 使用类似于 C 语言的语法结构来描述语法。

Functions are described by their names, which are constructed as syntax element names, with left and right round parentheses including zero or more variable names (for definition) or values (for usage), separated by commas (if more than one variable).

函数由它们的名称来描述，构造方式和语法元素的名称相同，圆括号内包含零个或多个变量名称或值，由逗号分开（如果多于一个变量）。

Square parentheses are used for indexing in lists or arrays. Lists or arrays can either be syntax elements or variables.

方括号被用来索引列表或者数组，列表或数组可能是语法元素或变量。

Binary notation is indicated by enclosing the string of bit values by single quote marks. For example, '01000001' represents an eight-bit string having only its second and its last bits equal to 1.

使用单引号来用二进制符号表示比特串的值。例如：'01000001' 表示一个 8 比特串，只有它的第二个和最后一个比特位等于 1。

Hexadecimal notation, indicated by prefixing the hexadecimal number by "0x", may be used instead of binary notation when the number of bits is a multiple of 4. For example, 0x41 represents an eight-bit string having only its second and its last bits equal to 1.

当比特的数目为 4 的整数倍时，那么使用前缀有十六进制符号"0x"的十六进制来替代二进制。例如：0x41 表示一个 8 比特串，只有第二个和最后一个比特位等于 1。

Numerical values not enclosed in single quotes and not prefixed by "0x" are decimal values.

不使用单引号并且没有前缀 "0x" 的数值是十进制值。

A value equal to 0 represents a FALSE condition in a test statement. The value TRUE is represented by any other value different than zero.

在检验申明中，等于 0 的值表示 FALSE 条件，值 TRUE 被表示成任何一个非零值。

5.9 Text description of logical operations

逻辑操作的文字描述

In the text, a statement of logical operations as would be described in pseudo-code as

在文中，逻辑操作申明的伪代码描述如下：

```
if( condition 0 )
  statement 0
else if ( condition 1 )
  statement 1
…
else /* informative remark on remaining condition */
  statement n
```

may be described in the following manner:

也许用用下面的方式描述

- If condition 0, statement 0
- If condition 1, statement 1
- …
- Otherwise (informative remark on remaining condition), statement n

In the text, a statement of logical operations as would be described in pseudo-code as

在文中，逻辑操作申明的伪代码描述如下：

```
if( condition 0a  &&  condition 0b )
  statement 0
else if ( condition 1a  ||  condition 1b )
  statement 1
…
else
  statement n
```

may be described in the following manner:

也许用用下面的方式描述

- If all of the following conditions are true, statement 0
- 如果下面所有的条件满足
  - condition 0a
  - condition 0b
- If any of the following conditions are true, statement 1
- 如果下面所有的条件满足
  - condition 1a

– condition 1b

– …

– Otherwise, statement n

In the text, a statement of logical operations as would be described in pseudo-code as

在文中，逻辑操作申明的伪代码描述如下：

```
if( condition 0 )
 statement 0
if ( condition 1 )
 statement 1
```

may be described in the following manner:

也许用用下面的方式描述

When condition 0, statement 0

When condition 1, statement 1

5.10 Processes

过程

Processes are used to describe the decoding of syntax elements. A process has a separate specification and invoking. All syntax elements and upper case variables that pertain to the current syntax structure and depending syntax structures are available in the process specification and invoking. A process specification may also have a lower case variable explicitly specified as the input. Each process specification has explicitly specified an output. The output is a variable that can either be an upper case variable or a lower case variable. If invoking a process, variables are explicitly assigned to lower case input or output variables of the process specification in case these do not have the same name. Otherwise (when the variables at the invoking and specification have the same name), assignment is implied.

过程被用来描述语法元素的解码。一个过程有一个单独的说明和调用。在过程的调用和说明的过程中，属于当前语法结构和相关语法结构的所有语法元素和大写字母变量都是可用的。过程的说明中也许有可用的小写字母变量作为输入，每一个过程都有一个显式的说明输出，这个输出变量可能是大写字母变量，也可能是小写字母变量。如果调用一个过程中，万一没有相同名字的情况下，变量将被显式的分配给小写字母输入或者过程的输出变量，否则（当调用的变量和说明的变量有相同的名字），使用隐式的分配过程。

In the specification of a process, a specific macroblock may be referred to by the variable name having a value equal to the address of the specific macroblock.

在过程的说明过程中，具体的宏块可能使用对应宏块的地址作为变量名称。

6 Source, coded, decoded, output data formats, scanning processes, and neighbouring relationships

6.1 Bitstream formats

This subclause specifies the relationship between the NAL unit stream and byte stream, either of which are referred to as the bitstream.

本小节说明了字节流和 NAL 单元流之间的关系，它们都被称为比特流。

The bitstream can be in one of two formats: the NAL unit stream format or the byte stream format. The NAL unit stream format is conceptually the more "basic" type. It consists of a sequence of syntax structures called NAL units. This sequence is ordered in decoding order. There are constraints imposed on the decoding order (and contents) of the NAL units in the NAL unit stream.

比特流可以是这两种格式中的一种：NAL 单元流格式和字节流格式。NAL 单元流格式是概念上更加“基本”的类型，它由称为 NAL 单元的语法结构序列组成，这个序列按解码的顺序进行排列。在 NAL 单元流中，对 NAL 单元的解码顺序是有限制的。

The byte stream format can be constructed from the NAL unit stream format by ordering the NAL units in decoding order and prefixing each NAL unit with a start code prefix and zero or more zero-valued bytes to form a stream of bytes.

The NAL unit stream format can be extracted from the byte stream format by searching for the location of the unique start code prefix pattern within this stream of bytes. Methods of framing the NAL units in a manner other than use of the byte stream format are outside the scope of this Recommendation | International Standard. The byte stream format is specified in Annex B.

字节流格式由 NAL 单元流来构造，按解码顺序对 NAL 单元流进行排序并且在每个 NAL 单元流之前加开始码和一个或者多个零字节来形成字节流。在字节流中，可以通过搜索唯一的开始码，来将 NAL 单元流从字节流中分离出来。除了使用字节流的格式来构造 NAL 单元流之外，构造 NAL 单元流的方法不是本协议的内容。字节流格式在附录 B 中说明。

6.2 Source, decoded, and output picture formats

6.2 源，解码和输出图像格式

This subclause specifies the relationship between source and decoded frames and fields that is given via the bitstream.

本小节说明了源和解码帧与场之间的关系，它们都由码流给定。

The video source that is represented by the bitstream is a sequence of either or both frames or fields (called collectively pictures) in decoding order.

由码流表示的视频源是按解码顺序的帧或场（被称为图像）的序列。

The source and decoded pictures (frames or fields) are each comprised of three sample arrays, one luma and two chroma sample arrays.

源和解码图像都由三个采样数组组成，一个亮度采样数组和两个色差采样数组。

The variable ChromaFormatFactor is specified in Table 6-1, depending on the chroma format sampling structure. The value of ChromaFormatFactor shall be inferred equal to 1.5, indicating 4:2:0 sampling. In monochrome sampling there is only one sample array, which may nominally be considered a luma array. In 4:2:0 sampling, each of the two chroma arrays has half the height and half the width of the luma array. In 4:2:2 sampling, each of the two chroma arrays has the same height and half the width of the luma array. In 4:4:4 sampling, each of the two chroma arrays has the same height and width as the luma array.

依据色差的采样结构，变量 ChromaFormatFactor 的说明如表 Table 6-1 所示，ChromaFormatFactor 的值等于 1.5 表示 4:2:0 采样。在单色采样中，只有一个采样数组，一般被称为亮度数组。在 4:2:0 采样中，有两个色差采样数组，每个色差采样数组的宽与高都是亮度数组宽与高的一半。在 4:2:2 采样中，也有两个色差数组，每个色差数组和亮度数组有相同的高，但宽只有亮度数组的一半。在 4:4:4 采样中，也有两个色差数组，每个色差数组的宽与高和亮度数组相同。

NOTE – Other values may be valid for future versions of this Recommendation | International Standard.

注 — 在将来的协议版本中，其他的值也许是正确的。

Table 6-1 – ChromaFormatFactor values

Chroma Format	ChromaFormatFactor
monochrome	1
4:2:0	1.5
4:2:2	2
4:4:4	3

The width and height of the luma sample array are each a multiple of 16. This Recommendation | International Standard represents colour sequences using 4:2:0 chroma sampling. The width and height of chroma sample arrays are each a multiple of 8. The height of a luma array that is coded as two separate fields or in macroblock-adaptive frame-field coding (see below) is a multiple of 32 samples. The height of each chroma array that is coded as two separate fields or in macroblock-adaptive frame-field coding (see below) is a multiple of 16 samples. The width or height of pictures output from the decoding process need not be a multiple of 16 and can be specified using a cropping rectangle.

亮度采样数组的宽与高是 16 的整数倍，本协议使用 4:2:0 色差采样格式来表示彩色序列。色差采样数组的宽与高应该是 8 的整数倍。如果按两个单独的场编码或者采用宏块自适应帧场编码方式，那么亮度采样数组的宽与高应该是 32 的整数倍，而色差采样数组的宽与高应该是 16 的整数倍。解码输出图像的宽与高不需要为 16 的整数倍，可以由裁减矩形来说明。

The width of fields coded referring to a specific sequence parameter set is the same as that of frames coded referring to the same sequence parameter set (see below). The height of fields coded referring to a specific sequence parameter set is half that of frames coded referring to the same sequence parameter set (see below).

对于一个具体的序列参数集来说，编码场的宽度和编码帧的宽度是相同的，而编码场的高度是编码帧的高度的一半。

The nominal vertical and horizontal relative locations of luma and chroma samples in frames are shown in Figure 6-1. Alternative chroma sample relative locations may be indicated in video usability information (see Annex E).

在图像帧中，亮度采样和色差采样的垂直相对位置和水平相对位置如图 Figure 6-1 所示，可选择的色差相对位置由视频可用信息（video usability information）表示（见附录 E）。

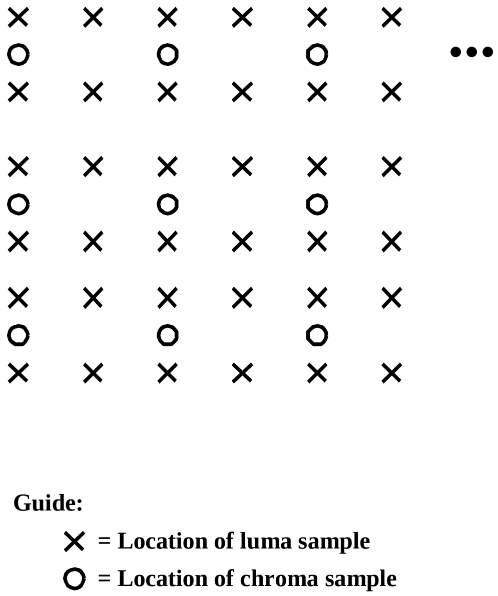


Figure 6-1 – Nominal vertical and horizontal locations of 4:2:0 luma and chroma samples in a frame

A frame consists of two fields as described below. A coded picture may represent a coded frame or an individual coded field. A coded video sequence conforming to this Recommendation | International Standard may contain arbitrary combinations of coded frames and coded fields. The decoding process is also specified in a manner that allows smaller regions of a coded frame to be coded either as a frame or field region, by use of macroblock-adaptive frame-field coding.

一帧由两场组成。一编码图像可能表示一编码帧，也可能表示一编码场。遵循本协议的编码视频序列可以包含编码帧和编码场的任意组合。通过使用宏块级帧场自适应编码，一编码帧可以按比编码帧或者编码场更小的区域进行帧场编码，解码过程也是按这种方式进行说明。

Source and decoded fields are one of two types: top field or bottom field. When two fields are output at the same time, or are combined to be used as a reference frame (see below), the two fields (which shall be of opposite parity) are interleaved. The first (i.e., top), third, fifth, etc. rows of a decoded frame are the top field rows. The second, fourth, sixth, etc. rows of a decoded frame are the bottom field rows. A top field consists of only the top field rows of a decoded frame. When the top field or bottom field of a decoded frame is used as a reference field (see below) only the even rows (for a top field) or the odd rows (for a bottom field) of the decoded frame are used.

源场和解码场有两种类型：顶场和底场。当这两场输出时间相同，或者被合并成一个参考帧时，那么这两场（这两场应该是相对的部分，如一场为顶场，那么另一场就应该是底场）应该是隔行存储。解码帧的第一、三、五等行应该是顶场的行，第二、四、六等行应该是底场的行。一顶场仅仅由一解码帧中顶场的行组成。当一解码帧的顶场或底场用作参考场时，那么仅仅使用该解码帧的偶数行（顶场）或奇数行（底场）。

The nominal vertical and horizontal relative locations of luma and chroma samples in top and bottom fields are shown in Figure 6-2. The nominal vertical sampling relative locations of the chroma samples in a top field are specified as shifted up by one-quarter luma sample height relative to the field-sampling grid. The vertical sampling locations of the chroma samples in a bottom field are specified as shifted down by one-quarter luma sample height relative to the field-sampling grid. Alternative chroma sample relative locations may be indicated in the video usability information (see Annex E).

在顶场和底场中，亮度采样和色差采样的相对水平位置和垂直位置如图 Figure 6-2 所示。在顶场中，色差采样的垂直采样相对位置向上移动相当于场采样网格的1/4亮度象素高度。在底场中，色差采样的垂直采样相对位置向下移动相当于场采样网格的1/4亮度象素高度。可选择的色差采样相对位置由视频可用信息来表示（见附录E）

NOTE – The shifting of the chroma samples is in order for these samples to align vertically to the usual location relative to the full-frame sampling grid as shown in Figure 6-1.

注 － 色差采样的移动时为了在全帧采样网格中（如图 Figure 6-1 ）使这些采样在可用的位置处垂直对齐。（这里可以画一个图来表示）

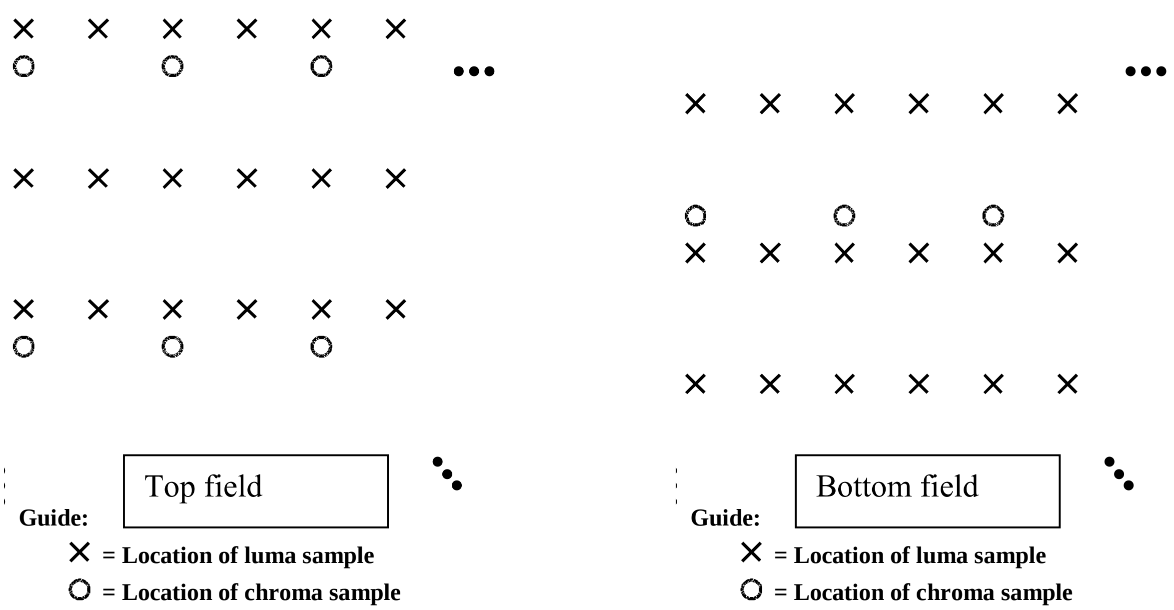


Figure 6-2 – Nominal vertical and horizontal sampling locations of samples top and bottom fields.

6.3 Spatial subdivision of pictures and slices

6.3 图像和 slice 的空间划分

This subclause specifies how a picture is partitioned into slices and macroblocks. Pictures are divided into slices. A slice is a sequence of macroblocks, or, when macroblock-adaptive frame/field decoding is in use, a sequence of macroblock pairs.

本小节说明了图像如何划分成 slice 和 宏快。图像被分成 slice， slice 是宏块的序列，或者当使用宏块帧场自适应解码时，slice 是宏块对的序列。

Each macroblock is comprised of one 16x16 luma and two 8x8 chroma sample arrays. When macroblock-adaptive frame/field decoding is not in use, each macroblock represents a spatial rectangular region of the picture. For example, a picture may be divided into two slices as shown in Figure 6-3.

每个宏块由一个 16x16 亮度数组和 两个 8x8 色差数组组成。当不使用宏块自适应解码时，每个宏块是图像的一个空间矩形区域。例如图像可能被分割成两个 slice，如图 Figure 6-3 。

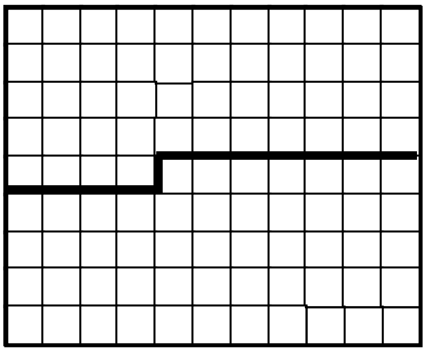


Figure 6-3 – A picture with 11 by 9 macroblocks that is partitioned into two slices

When macroblock-adaptive frame/field decoding is in use, the picture is partitioned into slices containing an integer number of macroblock pairs as shown in Figure 6-4. Each macroblock pair consists of two macroblocks.

当使用宏快帧场自适应解码时，图像可以被划分成包含整数个宏块对的 slice，如图 Figure 6-4 所示，每个宏块对由两个宏块组成。

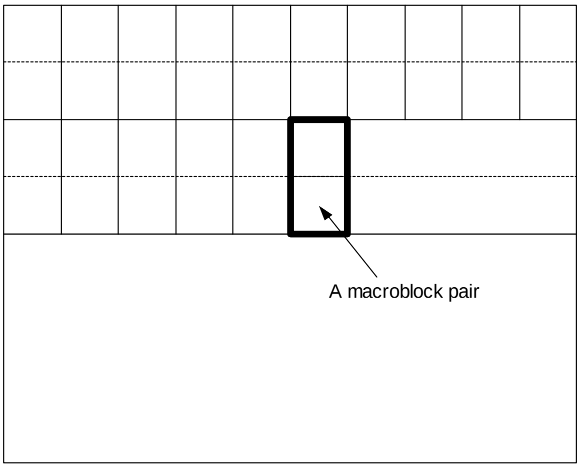


Figure 6-4 – Partitioning of the decoded frame into macroblock pairs.

6.4 Inverse scanning processes and derivation processes for neighbours

6.4 对于相邻的反向扫描过程和导出过程

This subclause specifies inverse scanning processes; i.e., the mapping of indices to locations, and derivation processes for neighbours.

本小节说明了反向扫描过程，例如：slice 位置的映射和相邻位置的导出。

6.4.1 Inverse macroblock scanning process

6.4.1 反向宏块扫描过程

Input to this process is a macroblock address mbAddr.

本过程的输入是宏块地址 mbAddr。

Output of this process is the location (x, y) of the upper-left luma sample for the macroblock with address mbAddr relative to the upper-left sample of the picture.

本过程的输出是地址为 mbAddr 的宏块左上角亮度象素相对于图像左上角象素的坐标位置 (x, y)。

The inverse macroblock scanning process is specified as follows.

方向宏块扫描过程如下所示：

- If MbaffFrameFlag is equal to 0,
- 如果 MbaffFrameFlag 等于 0，

$$x = \text{InverseRasterScan}(\ \text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 0\ ) \quad (6\text{-}1)$$

$$y = \text{InverseRasterScan}(\ \text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 1\ ) \quad (6\text{-}2)$$

- Otherwise (MbaffFrameFlag is equal to 1), the following applies.
- 否则（MbaffFrameFlag 等于 1），使用下面的过程。

$$xO = \text{InverseRasterScan}(\ \text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 0\ ) \quad (6\text{-}3)$$

$$yO = \text{InverseRasterScan}(\ \text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 1\ ) \quad (6\text{-}4)$$

  - If the current macroblock is a frame macroblock
  - 如果当前宏块是一帧宏块

$$x = xO \quad (6\text{-}5)$$

$$y = yO + (\ \text{mbAddr} \,\%\, 2\ ) * 16 \quad (6\text{-}6)$$

  - Otherwise (the current macroblock is a field macroblock),
  - 否则（当前宏块是一场宏块），

$$x = xO \quad (6\text{-}7)$$

$$y = yO + (\ \text{mbAddr} \,\%\, 2\ ) \quad (6\text{-}8)$$

6.4.2 Inverse macroblock partition and sub-macroblock partition scanning process

6.4.2 反向宏块部分和子宏块部分的扫描过程

Macroblocks or sub-macroblocks may be partitioned, and the partitions are scanned for inter prediction as shown in Figure 6-5. The outer rectangles refer to the samples in a macroblock or sub-macroblock, respectively. The rectangles refer to the partitions. The number in each rectangle specifies the index of the inverse macroblock partition scan or inverse sub-macroblock partition scan.

宏块或子宏块可以被分割，在帧间预测中，分割部分的扫描如图 Figure 6-5 所示。外部的矩形分别表示宏块或子宏块部分的象素，内部的矩形分别称为部分，每个矩形中的数字表示了反向宏块部分扫描或反向子宏块部分扫描的索引。

The functions MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ), and SubMbPartHeight( ) describing the width and height of macroblock partitions and sub-macroblock partitions are specified in Table 7-10, Table 7-11, Table 7-14, and Table 7-15. MbPartWidth( ) and MbPartHeight( ) are set to appropriate values for each macroblock, depending on the macroblock type. SubMbPartWidth( ) and SubMbPartHeight( ) are set to appropriate values for each sub-macroblock of a macroblocks with mb_type equal to P_8x8, P_8x8ref0, or B_8x8, depending on the sub-macroblock type.

函数 MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ), 和 SubMbPartHeight( )分别描述了宏块或子宏块部分的宽度和高度，如表 Table 7-10, Table 7-11, Table 7-14, 和 Table 7-15 所示。对于每一个宏块，依据宏块的类型，MbPartWidth( ) 和 MbPartHeight( ) 被设置成适当的值，对于 mb_type 等于 P_8x8, P_8x8ref0, 或 B_8x8 的宏块中的每一个子宏块来说，依据子宏块的类型，SubMbPartWidth( ) 和 SubMbPartHeight( ) 被设置成适当的值。

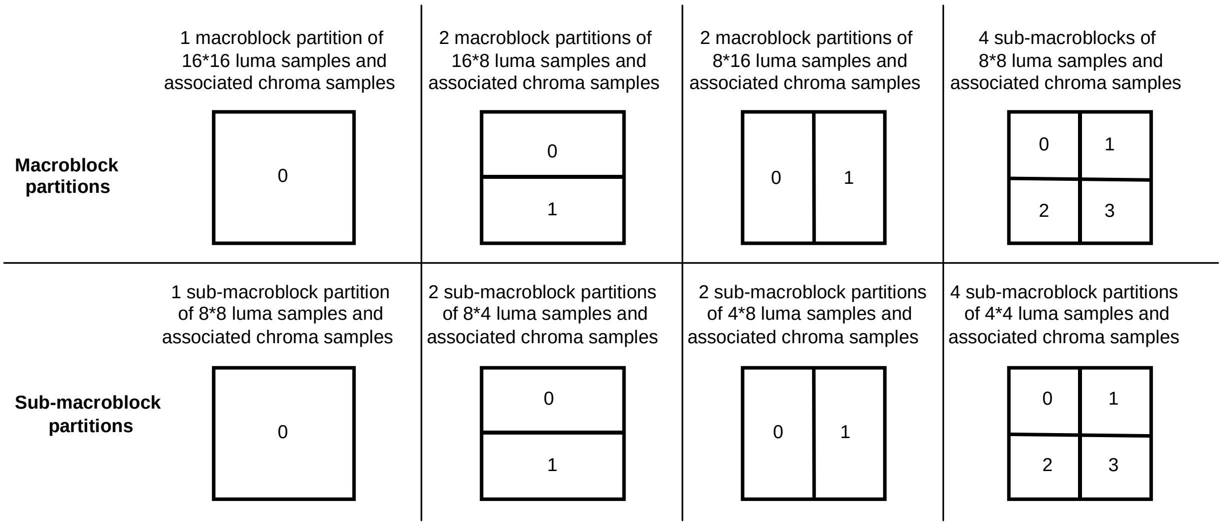


Figure 6-5 – Macroblock partitions, sub-macroblock partitions, macroblock partition scans, and sub-macroblock partition scans.

6.4.2.1 Inverse macroblock partition scanning process

6.4.2.1 反向宏块部分扫描过程

Input to this process is the index of a macroblock partition mbPartIdx.

本过程的输入是宏块部分的索引 mbPartIdx。

Output of this process is the location (x, y) of the upper-left luma sample for the macroblock partition mbPartIdx relative to the upper-left sample of the macroblock.

本过程的输出是索引为 mbPartIdx宏块部分的左上角象素相对于宏块的左上角象素的坐标位置 (x, y)。

The inverse macroblock partition scanning process is specified by

反向宏块部分扫描过程说明如下：

$$x = \text{InverseRasterScan}(\text{mbPartIdx}, \text{MbPartWidth}(\text{mb\_type}), \text{MbPartHeight}(\text{mb\_type}), 16, 0) \quad (6\text{-}9)$$

$$y = \text{InverseRasterScan}(\text{mbPartIdx}, \text{MbPartWidth}(\text{mb\_type}), \text{MbPartHeight}(\text{mb\_type}), 16, 1) \quad (6\text{-}10)$$

6.4.2.2 Inverse sub-macroblock partition scanning process

6.4.2.2 反向子宏块部分的扫描过程

Inputs to this process are the index of a macroblock partition mbPartIdx and the index of a sub-macroblock partition subMbPartIdx.

本过程的输入是宏块部分索引和子宏块部分索引。

Output of this process is the location (x, y) of the upper-left luma sample for the sub-macroblock partition subMbPartIdx relative to the upper-left sample of the sub-macroblock.

本过程的输出是索引为 subMbPartIdx 的子宏块部分的左上角象素相对于子宏块的左上角象素的坐标位置 (x, y)。

The inverse sub-macroblock partition scanning process is specified as follows.

反向子宏块部分的扫描过程说明如下：

- If mb_type is equal to P_8x8, P_8x8ref0, or B_8x8,

  如果mb_type 等于 P_8x8, P_8x8ref0, 或 B_8x8,

$$x = \text{InverseRasterScan(subMbPartIdx, SubMbPartWidth(sub\_mb\_type[mbPartIdx]), SubMbPartHeight(sub\_mb\_type[mbPartIdx]), 8, 0)} \quad (6\text{-}11)$$

$$y = \text{InverseRasterScan(subMbPartIdx, SubMbPartWidth(sub\_mb\_type[mbPartIdx]), SubMbPartHeight(sub\_mb\_type[mbPartIdx]), 8, 1)} \quad (6\text{-}12)$$

- Otherwise,

  否则

$$x = \text{InverseRasterScan(subMbPartIdx, 4, 4, 8, 0)} \quad (6\text{-}13)$$

$$y = \text{InverseRasterScan(subMbPartIdx, 4, 4, 8, 1)} \quad (6\text{-}14)$$

6.4.3 Inverse 4x4 luma block scanning process

反4x4亮度块扫描过程

Input to this process is the index of a 4x4 luma block luma4x4BlkIdx.

该过程的输入是4x4亮度块luma4x4BlkIdx的索引

Output of this process is the location (x, y) of the upper-left luma sample for the 4x4 luma block with index luma4x4BlkIdx relative to the upper-left luma sample of the macroblock.

该过程的输出是相对于宏块左上角样本索引为luma4x4BlkIdx的4x4亮度块的做上角样本点的位置(x, y)。

Figure 6-6 shows the scan for the 4x4 luma blocks.

图6-6表示了4x4亮度块的扫描。

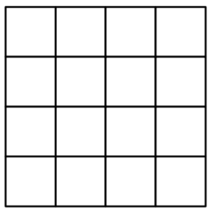


Figure 6-6 – Scan for 4x4 luma blocks.

The inverse 4x4 luma block scanning process is specified by

反4x4亮度块的扫描过程说明如下

$$x = \text{InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 0) + InverseRasterScan(luma4x4BlkIdx \% 4, 4, 4, 8, 0)} \quad (6\text{-}15)$$

$$y = \text{InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 1) + InverseRasterScan(luma4x4BlkIdx \% 4, 4, 4, 8, 1)} \quad (6\text{-}16)$$

6.4.4 Derivation process of the availability for macroblock addresses

导出某地址宏块的可用性的过程

Input to this process is a macroblock address mbAddr.

输入是宏块的地址mbAddr

Output of this process is the availability of the macroblock mbAddr.

输出是地址为mbAddr宏块的可用性

NOTE – The meaning of availability is determined when this process is invoked.
注—该过程调用时可以确定可用性

The macroblock is marked as available, unless one of the following conditions is true in which case the macroblock shall be marked as not available:

若宏块地址满足以下条件中任意一条，宏块标志为不可用，否则标志为可用。

- mbAddr < 0
- mbAddr > CurrMbAddr
- the macroblock with address mbAddr belongs to a different slice than the current slice
- 该地址的宏块属于的slice不是当前的slice

6.4.5 Derivation process for neighbouring macroblock addresses and their availability

6.4.5 相邻宏块地址和它们可用性的导出过程

This process can only be invoked when MbaffFrameFlag is equal to 0.

该过程仅当MbaffFrameFlag=0时调用。

The outputs of this process are

该过程的输出

- mbAddrA: the address and availability status of the macroblock to the left of the current macroblock.
- 当前宏块左边宏块的地址和可用性
- mbAddrB: the address and availability status of the macroblock above the current macroblock.
- 当前宏块上边宏块的地址和可用性
- mbAddrC: the address and availability status of the macroblock above-right of the current macroblock.
- 当前宏块右上角宏块的地址和可用性
- mbAddrD: the address and availability status of the macroblock above-left of the current macroblock.
- 当前宏块左上角宏块的地址和可用性

Figure 6-7 shows the relative spatial locations of the macroblocks with mbAddrA, mbAddrB, mbAddrC, and mbAddrD relative to the current macroblock with CurrMbAddr.

图6-7表示当前宏块CurrMbAddr与mbAddrA、 mbAddrB、 mbAddrC和 mbAddrD相对的空间位置。

mbAddrD	mbAddrB	mbAddrC
mbAddrA	CurrMbAddr	

Figure 6-7 – Neighbouring macroblocks for a given macroblock

Input to the process in subclause 6.4.4 is mbAddrA = CurrMbAddr – 1 and the output is whether the macroblock mbAddrA is available. In addition, mbAddrA is marked as not available when CurrMbAddr % PicWidthInMbs is equal to 0.

小节6.4.4中的输入是mbAddrA = CurrMbAddr – 1，输出是块mbAddrA是否可用。此外，当CurrMbAddr % PicWidthInMbs =0时，mbAddrA被标志为不可用。

Input to the process in subclause 6.4.4 is mbAddrB = CurrMbAddr – PicWidthInMbs and the output is whether the macroblock mbAddrB is available.

小节6.4.4中的输入是mbAddrB = CurrMbAddr – PicWidthInMbs，输出是块mbAddrB是否可用。

Input to the process in subclause 6.4.4 is mbAddrC = CurrMbAddr – PicWidthInMbs + 1 and the output is whether the macroblock mbAddrC is available. In addition, mbAddrC is marked as not available when (CurrMbAddr + 1) % PicWidthInMbs is equal to 0.

小节6.4.4中的输入是mbAddrC = CurrMbAddr – PicWidthInMbs + 1， 输出是块mbAddrC是否可用。此外， 当(CurrMbAddr + 1) % PicWidthInMbs =0时，mbAddrA被标志为不可用。

Input to the process in subclause 6.4.4 is mbAddrD = CurrMbAddr – PicWidthInMbs - 1 and the output is whether the macroblock mbAddrD is available. In addition, mbAddrD is marked as not available when CurrMbAddr % PicWidthInMbs is equal to 0.

小节6.4.4中的输入是mbAddrD = CurrMbAddr – PicWidthInMbs - 1， 输出是块mbAddrD是否可用。此外， 当CurrMbAddr % PicWidthInMbs =0时，mbAddrA被标志为不可用。

6.4.6 Derivation process for neighbouring macroblock addresses and their availability in MBAFF frames

6.4.6 帧场自适应宏块帧中的相邻宏块地址和它们可用性的导出过程

This process can only be invoked when MbaffFrameFlag is equal to 1.

仅当MbaffFrameFlag=1时调用该过程。

The outputs of this process are

该过程的输出

- mbAddrA: the address and availability status of the top macroblock of the macroblock pair to the left of the current macroblock pair.
- 当前宏块对左边宏块对中顶宏块的地址和可用性状态
- mbAddrB: the address and availability status of the top macroblock of the macroblock pair above the current macroblock pair.
- 当前宏块对上边宏块对中顶宏块的地址和可用性状态
- mbAddrC: the address and availability status of the top macroblock of the macroblock pair above-right of the current macroblock pair.
- 当前宏块对右上角宏块对中顶宏块的地址和可用性状态
- mbAddrD: the address and availability status of the top macroblock of the macroblock pair above-left of the current macroblock pair.
- 当前宏块对左上角宏块对中顶宏块的地址和可用性状态

Figure 6-8 shows the relative spatial locations of the macroblocks with mbAddrA, mbAddrB, mbAddrC, and mbAddrD relative to the current macroblock with CurrMbAddr.

图6-8表示当前宏块CurrMbAddr与mbAddrA、 mbAddrB、 mbAddrC和 mbAddrD宏块相对的空间位置。

mbAddrA, mbAddrB, mbAddrC, and mbAddrD have identical values regardless whether the current macroblock is the top or the bottom macroblock of a macroblock pair.

无论当前宏块是宏块对的顶宏块或底宏块，mbAddrA、 mbAddrB、 mbAddrC和 mbAddrD都取同样的值。

mbAddrD	mbAddrB	mbAddrC
mbAddrA	CurrMbAddr or	
	CurrMbAddr	

Figure 6-8 – Neighbouring macroblocks for a given macroblock in MBAFF frames

Input to the process in subclause 6.4.4 is mbAddrA =2 * (CurrMbAddr / 2 – 1) and the output is whether the macroblock mbAddrA is available. In addition, mbAddrA is marked as not available when (CurrMbAddr / 2) % PicWidthInMbs is equal to 0.

小 节 6.4.4 中 的 输 入 是 mbAddrA =2 * (CurrMbAddr / 2 – 1)， 输 出 是 宏 块 mbAddrA 是 否 可 用 。 此 外， 当 (CurrMbAddr / 2) % PicWidthInMbs=0时，mbAddrA被标志为不可用。

Input to the process in subclause 6.4.4 is mbAddrB =2 * (CurrMbAddr / 2 – PicWidthInMbs) and the output is whether the macroblock mbAddrB is available.

小节6.4.4中的输入是mbAddrB =2 * (CurrMbAddr / 2 – PicWidthInMbs)，输出是宏块mbAddrB是否可用。

Input to the process in subclause 6.4.4 is mbAddrC = 2 * (CurrMbAddr / 2 – PicWidthInMbs + 1) and the output is whether the macroblock mbAddrC is available. In addition, mbAddrC is marked as not available when (CurrMbAddr / 2 + 1) % PicWidthInMbs is equal to 0.

小节6.4.4中的输入是mbAddrC = 2 * (CurrMbAddr / 2 – PicWidthInMbs + 1)，输出是宏块mbAddrC是否可用。此外，当(CurrMbAddr / 2 + 1) % PicWidthInMbs =0时，mbAddrA被标志为不可用。

Input to the process in subclause 6.4.4 is mbAddrD = 2 * (CurrMbAddr / 2 – PicWidthInMbs - 1) and the output is whether the macroblock mbAddrD is available. In addition, mbAddrD is marked as not available when (CurrMbAddr / 2) % PicWidthInMbs is equal to 0.

小节6.4.4中的输入是mbAddrD = 2 * (CurrMbAddr / 2 – PicWidthInMbs - 1)，输出是宏块mbAddrD是否可用。此外，当(CurrMbAddr / 2) % PicWidthInMbs =0时，mbAddrA被标志为不可用。

6.4.7 Derivation processes for neighbouring macroblocks, blocks, and partitions

6.4.7 相邻宏块、块和部分的导出过程

Subclause 6.4.7.1 specifies the derivation process for neighbouring macroblocks.

小节6.4.7.1说明相邻宏块的导出过程。

Subclause 6.4.7.2 specifies the derivation process for neighbouring 8x8 luma blocks.

小节6.4.7.2说明相邻8x8亮度块的导出过程。

Subclause 6.4.7.3 specifies the derivation process for neighbouring 4x4 luma blocks.

小节6.4.7.3说明相邻4x4亮度块的导出过程。

Subclause 6.4.7.4 specifies the derivation process for neighbouring 4x4 chroma blocks.

小节6.4.7.4说明相邻4x4色差块的导出过程

Subclause 6.4.7.5 specifies the derivation process for neighbouring partitions.

小节6.4.7.5说明相邻部分的导出过程。

Table 6-2 specifies the values for the difference of luma location (xD, yD) for the input and the replacement for N in mbAddrN, mbPartIdxN, subMbPartIdxN, luma8x8BlkIdxN, luma4x4BlkIdxN, and chroma4x4BlkIdxN for the output. These input and output assignments are used in subclauses 6.4.7.1 to 6.4.7.5. The variable predPartWidth is specified when Table 6-2 is referred to.

表6-2说明作为输入的亮度位置(xD, yD)的不同差值以及作为输出的N，N替代了mbAddrN、 mbPartIdxN、subMbPartIdxN、 luma8x8BlkIdxN、 luma4x4BlkIdxN和 chroma4x4BlkIdxN。这些输入和输出的赋值在小节6.4.7.1 到 6.4.7.5中使用。当表6-2被参考时，会对变量predPartWidth进行说明。

Table 6-2 – Specification of input and output assignments for subclauses 6.4.7.1 to 6.4.7.5

N	XD	yD
A	-1	0
B	0	-1
C	predPartWidth	-1
D	-1	-1

Figure 6-9 illustrates the relative location of the neighbouring macroblocks, blocks, or partitions A, B, C, and D to the current macroblock, partition, or block, when the current macroblock, partition, or block is in frame coding mode.

图6-9说明了在帧编码模式下当前宏块、块或部分的相邻宏块、块或部分A、B、C和 D的空间位置。

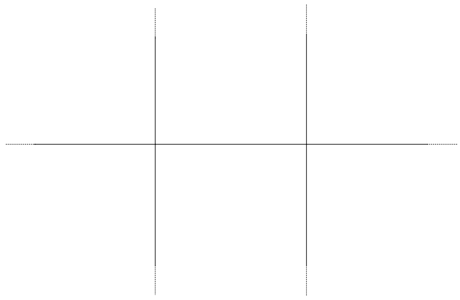


Figure 6-9 – Determination of the neighbouring macroblock, blocks, and partitions (informative)

6.4.7.1 Derivation process for neighbouring macroblocks

6.4.7.1 导出相邻宏块的过程

Outputs of this process are

该过程的输出

- mbAddrA: the address of the macroblock to the left of the current macroblock and its availability status and
- 当前宏块左边宏块的地址和它的可用性
- mbAddrB: the address of the macroblock above the current macroblock and its availability status.
- 当前宏块上边宏块的地址和它的可用性

mbAddrN (with N being A or B) is derived as follows.

mbAddrN (N 取 A 或 B)导出如下

- The difference of luma location (xD, yD) is set according to Table 6-2.
- 根据表6-2设置亮度位置(xD, yD)的差值。 不懂？
- The derivation process for neighbouring locations as specified in subclause 6.4.8 is invoked for luma locations with (xN, yN) equal to (xD, yD), and the output is assigned to mbAddrN.
- 对(xN, yN)= (xD, yD)的亮度位置调用相邻位置（在小节6.4.8中说明）的导出过程，输出赋值给mbAddrN。

6.4.7.2 Derivation process for neighbouring 8x8 luma block

6.4.7.2 相邻8x8亮度块的导出过程

Input to this process is an 8x8 luma block index luma8x8BlkIdx.

该过程的输入是8x8亮度块的索引luma8x8BlkIdx。

The luma8x8BlkIdx specifies the 8x8 luma blocks of a macroblock in a raster scan.

luma8x8BlkIdx指定按栅格扫描的宏块的8x8亮度块

Outputs of this process are

该过程的输出

- mbAddrA: either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock and its availability status,
- 等于CurrMbAddr或者等于当前宏块左边宏块的地址，以及它的可用状态。
- luma8x8BlkIdxA: the index of the 8x8 luma block to the left of the 8x8 block with index luma8x8BlkIdx and its availability status,

- 索引为luma8x8BlkIdx的8x8块左边8x8亮度块的索引，以及它的可用状态。
- mbAddrB: either equal to CurrMbAddr or the address of the macroblock above the current macroblock and its availability status,
- 等于CurrMbAddr或者等于当前宏块上边宏块的地址，以及它的可用状态。
- luma8x8BlkIdxB: the index of the 8x8 luma block above the 8x8 block with index luma8x8BlkIdx and its availability status.
- 索引为luma8x8BlkIdx的8x8块上边8x8亮度块的索引，以及它的可用状态。

mbAddrN and luma8x8BlkIdxN (with N being A or B) are derived as follows.

mbAddrN 和 luma8x8BlkIdxN (N 取A 或 B)导出如下。

- The difference of luma location (xD, yD) is set according to Table 6-2.
- 根据表6-2设置亮度位置(xD, yD)的差值。
- The luma location (xN, yN) is specified by
- 亮度位置(xN, yN)的说明如下

$$xN = (luma8x8BlkIdx \% 2) * 8 + xD \quad (6-17)$$

$$yN = (luma8x8BlkIdx / 2) * 8 + yD \quad (6-18)$$

- The derivation process for neighbouring locations as specified in subclause 6.4.8 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- 为亮度位置调用相邻位置（小节6.4.8中说明）的导出过程，(xN, yN)作为输入，输出赋值给mbAddrN和(xW, yW)
- If mbAddrN is not available, luma8x8BlkIdxN is marked as not available.
- 如果mbAddrN不可用，luma8x8BlkIdxN标志为不可用。
- Otherwise (mbAddrN is available), the 8x8 luma block in the macroblock mbAddrN covering the luma location (xW, yW) shall be assigned to luma8x8BlkIdxN.
- 否则（mbAddrN可用），宏块mbAddrN中覆盖亮度位置(xW, yW)的8x8的亮度块赋值为luma8x8BlkIdxN。

6.4.7.3 Derivation process for neighbouring 4x4 luma blocks

6.4.7.3 相邻4x4亮度块的导出过程

Input to this process is a 4x4 luma block index luma4x4BlkIdx.

该过程的输入是索引为luma4x4BlkIdx的4x4亮度块

Outputs of this process are

该过程的输出

- mbAddrA: either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock and its availability status,
- 等于CurrMbAddr或者等于当前宏块左边宏块的地址，以及它的可用状态，
- luma4x4BlkIdxA: the index of the 4x4 luma block to the left of the 4x4 block with index luma4x4BlkIdx and its availability status,
- 等于索引为luma4x4BlkIdx的4x4块左边4x4亮度块的索引，以及它的可用状态。
- mbAddrB: either equal to CurrMbAddr or the address of the macroblock above the current macroblock and its availability status,
- 等于CurrMbAddr或者等于当前宏块上边宏块的地址，以及它的可用状态，
- luma4x4BlkIdxB: the index of the 4x4 luma block above the 4x4 block with index luma4x4BlkIdx and its availability status.

- 等于索引为luma4x4BlkIdx的4x4块上边4x4亮度块的索引，以及它的可用状态。

mbAddrN and luma4x4BlkIdxN (with N being A or B) are derived as follows.

mbAddrN 和luma4x4BlkIdxN (N取A或B)导出如下

- The difference of luma location (xD, yD) is set according to Table 6-2.
- 根据表6-2设置亮度位置(xD, yD)的差值
- The inverse 4x4 luma block scanning process as specified in subclause 6.4.3 is invoked with luma4x4BlkIdx as the input and (x, y) as the output.
- 调用反4x4亮度块扫描过程（小节6.4.3中），luma4x4BlkIdx作为输入，(x, y)作为输出。
- The luma location (xN, yN) is specified by
- 亮度位置(xN, yN)指定为

$$xN = x + xD \tag{6-19}$$

$$yN = y + yD \tag{6-20}$$

- The derivation process for neighbouring locations as specified in subclause 6.4.8 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- 对亮度位置调用该过程导出相邻位置（小节6.4.8中说明），以(xN, yN)作为输入，输出赋值给mbAddrN和(xW, yW)
- If mbAddrN is not available, luma4x4BlkIdxN is marked as not available.
- 如果mbAddrN不可用，luma4x4BlkIdxN标志为不可用。
- Otherwise (mbAddrN is available), the 4x4 luma block in the macroblock mbAddrN covering the luma location (xW, yW) shall be assigned to luma4x4BlkIdxN.
- 否则（mbAddrN可用），覆盖亮度位置(xW, yW)的宏块mbAddrN中的4x4亮度块赋值给luma4x4BlkIdxN。

6.4.7.4 Derivation process for neighbouring 4x4 chroma blocks

6.4.7.4 相邻4x4色差块得导出过程

Input to this is a current 4x4 chroma block chroma4x4BlkIdx.

该过程的输入是当前4x4色差块chroma4x4BlkIdx。

Outputs of this process are

过程的输出是

- mbAddrA: either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock and its availability status,
- 等于CurrMbAddr或者等于当前宏块左边宏块的地址，以及它的可用状态。
- chroma4x4BlkIdxA: the index of the 4x4 chroma block to the left of the chroma 4x4 block with index chroma4x4BlkIdx and its availability status,
- 索引为chroma4x4BlkIdx的色差4x4块左边4x4色差块的索引，以及它的可用状态。
- mbAddrB: either equal to CurrMbAddr or the address of the macroblock above the current macroblock and its availability status,
- 等于CurrMbAddr或者当前宏块上边宏块的地址，以及它的可用状态。
- chroma4x4BlkIdxB: the index of the 4x4 chroma block above the chroma 4x4 block index chroma4x4BlkIdx and its availability status.
- 索引为chroma4x4BlkIdx的色差4x4块上边4x4色差块的索引，以及它的可用状态。

The derivation process for neighbouring 8x8 luma block is invoked with luma8x8BlkIdx = chroma4x4BlkIdx as the input and with mbAddrA, chroma4x4BlkIdxA = luma8x8BlkIdxA, mbAddrB, and chroma4x4BlkIdxB = luma8x8BlkIdxB as the output.

为相邻位置调用该导出过程，luma8x8BlkIdx = chroma4x4BlkIdx作为输入，mbAddrA、chroma4x4BlkIdxA = luma8x8BlkIdxA、mbAddrB、和chroma4x4BlkIdxB = luma8x8BlkIdxB作为输出。

6.4.7.5 Derivation process for neighbouring partitions

6.4.7.5 相邻部分的导出过程

Inputs to this process are

该过程的输入

- a macroblock partition index mbPartIdx
- 索引mbPartIdx的宏块部分
- a sub-macroblock partition index subMbPartIdx
- 索引为subMbPartIdx的子宏块部分

Outputs of this process are

过程的输出

- mbAddrA\mbPartIdxA\subMbPartIdxA: specifying the macroblock or sub-macroblock partition to the left of the current macroblock and its availability status, or the sub-macroblock partition CurrMbAddr\mbPartIdx\subMbPartIdx and its availability status,
- 指定当前宏块左边宏块或子宏块部分以及它的可用状态
- mbAddrB\mbPartIdxB\subMbPartIdxB: specifying the macroblock or sub-macroblock partition above the current macroblock and its availability status, or the sub-macroblock partition CurrMbAddr\mbPartIdx\subMbPartIdx and its availability status,
- 指定当前宏块上边的宏块或子宏块部分及其可用状态，或者子宏块部分CurrMbAddr\mbPartIdx\subMbPartIdx及其可用状态
- mbAddrC\mbPartIdxC\subMbPartIdxC: specifying the macroblock or sub-macroblock partition to the right-above of the current macroblock and its availability status, or the sub-macroblock partition CurrMbAddr\mbPartIdx\subMbPartIdx and its availability status,
- 指定当前宏块右上角宏块或子宏块部分及其可用状态，或者是子宏块部分CurrMbAddr\mbPartIdx\subMbPartIdx及其可用状态
- mbAddrD\mbPartIdxD\subMbPartIdxD: specifying the macroblock or sub-macroblock partition to the left-above of the current macroblock and its availability status, or the sub-macroblock partition CurrMbAddr\mbPartIdx\subMbPartIdx and its availability status.
- mbAddrD\mbPartIdxD\subMbPartIdxD ：指定当前宏块左上角宏块或子宏块部分及其可用状态，或者是子宏块部分CurrMbAddr\mbPartIdx\subMbPartIdx及其可用状态。

mbAddrN, mbPartIdxN, and subMbPartIdx (with N being A, B, C, or D) are derived as follows.

MbAddrN、 mbPartIdxN和subMbPartIdx (N 取值A、B、C或D)生成如下

- The inverse macroblock partition scanning process as described in subclause 6.4.2.1 is invoked with mbPartIdx as the input and (x, y) as the output.
- 调用小节6.4.2.1中描述的逆向宏块部分扫描过程，mbPartIdx作为输入，(x, y)作为输出
- The location of the upper-left luma sample inside a macroblock partition (xS, yS) is derived as follows.
- 宏块部分(xS, yS)中左上角亮度像素位置生成如下：
  - If mb_type is equal to P_8x8, P_8x8ref0 or B_8x8, the inverse sub-macroblock partition scanning process as described in subclause 6.4.2.2 is invoked with subMbPartIdx as the input and (xS, yS) as the output.
  - 如果mb_type等于P_8x8、P_8x8ref0或者B_8x8，调用小节6.4.2.2中描述的逆向子宏块部分扫描过程，以subMbPartIdx作为输入，(xS, yS)作为输出。
  - Otherwise, (xS, yS) are set to (0, 0).
  - 否则，(xS, yS)设为(0, 0)

- The variable predPartWidth in Table 6-2 is specified as follows.
- 表6-2中变量predPartWidth的说明如下：
  - If mb_type is equal to P_Skip or B_Skip, or mb_type is equal to B_8x8 and sub_mb_type[mbPartIdx] is equal to B_Direct_8x8, predPartWidth = 16.
  - 如果mb_type等于P_Skip或B_Skip，或者mb_type等于B_8x8 并且sub_mb_type[mbPartIdx]等于B_Direct_8x8，那么predPartWidth = 16

  NOTE – When sub_mb_type[mbPartIdx] is equal to B_Direct_8x8, the predicted motion vector is the predicted motion vector for the complete macroblock independent of the value of mbPartIdx.

  注- 当sub_mb_type[mbPartIdx]等于B_Direct_8x8时，预测的运动矢量就是对整个宏块的预测运动矢量，不依赖于mbPartIdx的取值

  - If mb_type is equal to P_8x8, P_8x8ref0, or B_8x8 (and sub_mb_type[mbPartIdx] is not equal to B_Direct_8x8), predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]).
  - 如果mb_type等于P_8x8、P_8x8ref0或B_8x8（且sub_mb_type[mbPartIdx]不等于B_Direct_8x8)，那么predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx])。
  - Otherwise, predPartWidth = MbPartWidth(mb_type).
  - 否则，predPartWidth = MbPartWidth(mb_type)。
- The difference of luma location (xD, yD) is set according to Table 6-2.
- 根据表6-2设置亮度位置(xD, yD)的差值
- The neighbouring luma location (xN, yN) is specified by
- 相邻亮度位置(xN, yN)说明如下

$$xN = x + xS + xD \quad (6\text{-}21)$$

$$yN = y + yS + yD \quad (6\text{-}22)$$

- The derivation process for neighbouring locations as specified in subclause 6.4.8 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- 调用小节6.4.8说明的相邻位置导出过程，以(xN, yN)作为输入，输出赋值给mbAddrN和(xW, yW)
- If mbAddrN is not available, the macroblock or sub-macroblock partition mbAddrN\mbPartIdxN\subMbPartIdxN is marked as not available.
- 如果mbAddrN不可用，宏块或子宏块部分mbAddrN\mbPartIdxN\subMbPartIdxN标志为不可用。
- Otherwise (mbAddrN is available), the following applies.
- 否则（mbAddrN可用），应用下面的过程
  - The macroblock partition in the macroblock mbAddrN covering the luma location (xW, yW) shall be assigned to mbPartIdxN and the sub-macroblock partition inside the macroblock partition mbPartIdxN covering the sample (xW, yW) in the macroblock mbAddrN shall be assigned to subMbPartIdxN.
  - 宏块mbAddrN中覆盖亮度位置(xW, yW)的宏块部分赋值为mbPartIdxN，宏块mbAddrN中宏块部分mbPartIdxN内部覆盖像素(xW, yW)的子宏块部分赋值给subMbPartIdxN。
  - When the partition given by mbPartIdxN and subMbPartIdxN is not yet decoded, the macroblock partition mbPartIdxN and the sub-macroblock partition subMbPartIdxN are marked as not available.
  - 当由mbPartIdxN和subMbPartIdxN给定的部分尚未解码时，宏块部分mbPartIdxN和子宏块部分subMbPartIdxN标志为不可用。

    NOTE - The latter condition is, for example, the case when mbPartIdx = 2, subMbPartIdx = 3, xD = 4, yD = -1, i.e., when neighbour C of the last 4x4 luma block of the third sub-macroblock is requested.

    注- 后一个条件，就比如mbPartIdx = 2，subMbPartIdx = 3， xD = 4，yD = -1的情况，也就是需要第三个子宏块中最后一个4x4块的相邻块C的时候。

6.4.8 Derivation process for neighbouring locations

6.4.8 相邻位置的导出过程

Input to this process is a luma or chroma location (xN, yN) expressed relative to the upper left corner of the current macroblock

该过程的输入是相对于当前宏块左上角表示的亮度或色差位置(xN, yN)

Outputs of this process are

过程的输出是

- mbAddrN: either equal to CurrMbAddr or to the address of neighbouring macroblock that contains (xN, yN) and its availability status,
- 等于CurrMbAddr或者等于包含位置(xN, yN)的相邻宏块的地址以及它的可用状态。
- (xW, yW): the location (xN, yN) expressed relative to the upper-left corner of the macroblock mbAddrN (rather than relative to the upper-left corner of the current macroblock).
- 相对于宏块mbAddrN左上角（而不是相对于当前宏块的左上角）表示的位置(xN, yN)。

Let maxWH be a variable specifying a maximum value of the location components xN, yN, xW, and yW. maxWH is derived as follows.

令变量maxWH表示位置分量xN、yN、xW和yW的最大值。MaxWH生成如下

- If this process is invoked for neighbouring luma locations,
- 如果该过程是为了相邻的亮度位置调用

$$\text{maxWH} = 16 \tag{6-23}$$

- Otherwise (this process is invoked for neighbouring chroma locations),
- 否则（该过程是为了相邻的色差位置调用）

$$\text{maxWH} = 8 \tag{6-24}$$

Depending on the variable MbaffFrameFlag, the neighbouring luma locations are derived as follows

依赖变量MbaffFrameFlag，相邻亮度位置如下生成

- If MbaffFrameFlag is equal to 0, the specification for neighbouring luma locations in fields and non-MBAFF frames as described in subclause 6.4.8.1 is applied.
- 如果MbaffFrameFlag=0，应用小节6.4.8.1中描述的关于场和非帧场自适应宏块帧的相邻亮度位置的说明。
- Otherwise (MbaffFrameFlag is equal to 1), the specification for neighbouring luma locations in MBAFF frames as described in subclause 6.4.8.2 is applied.
- 否则（MbaffFrameFlag=1），应用6.4.8.2中描述的关于帧场自适应宏块帧中相邻亮度位置的说明。

6.4.8.1 Specification for neighbouring luma locations in fields and non-MBAFF frames

6.4.8.1 场和非帧场自适应宏块帧的相邻亮度位置的说明

The specifications in this subclause are applied when MbaffFrameFlag is equal to 0.

该小节中的说明应用于MbaffFrameFlag=0的情况。

The derivation process for neighbouring macroblock addresses and their availability in subclause 6.4.5 is invoked with mbAddrA, mbAddrB, mbAddrC, and mbAddrD as well as their availability status as the output.

调用小节6.4.5中说明的相邻宏块地址及其可用性的导出过程，mbAddrA、mbAddrB、mbAddrC和mbAddrD及其可用状态作为输出。

Table 6-3 specifies mbAddrN depending on (xN, yN).

表6-3说明依赖于(xN, yN)的mbAddrN的取值

Table 6-3 – Specification of mbAddrN

xN	yN	mbAddrN

< 0	< 0	mbAddrD
< 0	0 .. maxWH - 1	mbAddrA
0 .. maxWH - 1	< 0	mbAddrB
0 .. maxWH - 1	0 .. maxWH - 1	CurrMbAddr
> maxWH - 1	< 0	mbAddrC
> maxWH - 1	0 .. maxWH - 1	not available
	> maxWH - 1	not available

The neighbouring luma location (xW, yW) relative to the upper-left corner of the macroblock mbAddrN is derived as

相对于当前宏块mbAddrN相邻亮度位置(xW, yW)生成如下

$$xW = (xN + maxWH) \% maxWH \quad (6\text{-}25)$$

$$yW = (yN + maxWH) \% maxWH \quad (6\text{-}26)$$

6.4.8.2 Specification for neighbouring luma locations in MBAFF frames

6.4.8.2 帧场自适应宏块帧中相邻亮度位置的说明

The specifications in this subclause are applied when MbaffFrameFlag is equal to 1.

该小节中的说明应用于MbaffFrameFlag=0的情况

The derivation process for neighbouring macroblock addresses and their availability in subclause 6.4.6 is invoked with mbAddrA, mbAddrB, mbAddrC, and mbAddrD as well as their availability status as the output.

调用小节6.4.6中的相邻宏块地址及其可用性的生成过程，以mbAddrA、mbAddrB、mbAddrC和mbAddrD及其可用状态作为输出。

Table 6-4 specifies the macroblock addresses mbAddrN and yM in two ordered steps:

表6-4按两个有序的步骤说明宏块地址mbAddrN和yM：

1. Specification of a macroblock address mbAddrX depending on (xN, yN) and the following variables:

1. 依赖于(xN, yN)的宏块地址mbAddrX和以下变量说明如下：

   - The variable currMbFrameFlag is derived as follows.
   - 变量currMbFrameFlag生成如下
     - If the macroblock with address CurrMbAddr is a frame macroblock, currMbFrameFlag is set equal to 1,
     - 如果地址为CurrMbAddr的宏块是帧宏块，则将currMbFrameFlag设为1，
     - Otherwise (the macroblock with address CurrMbAddr is a field macroblock), currMbFrameFlag is set equal to 0.
     - 否则（地址为CurrMbAddr的宏块是场宏块），则将currMbFrameFlag设为0。
   - The variable mbIsTopMbFlag is derived as follows.
   - 变量mbIsTopMbFlag生成如下
     - If the macroblock with address CurrMbAddr is a top macroblock (CurrMbAddr % 2 is equal to 0), mbIsTopMbFlag is set equal to 1;
     - 如果地址为CurrMbAddr的宏块是顶宏块（CurrMbAddr % 2=0），mbIsTopMbFlag设置为1；
     - Otherwise (the macroblock with address CurrMbAddr is a bottom macroblock, CurrMbAddr % 2 is equal to 1), mbIsTopMbFlag is set equal to 0.
     - 否则（地址为CurrMbAddr的宏块是底宏块，CurrMbAddr % 2=1），mbIsTopMbFlag设置为0。

2. Depending on the availability of mbAddrX, the following applies.

2. 依赖mbAddrX的可用性，应用如下描述

- If mbAddrX is not available, mbAddrN is marked as not available.
- 如果mbAddrX不可用，mbAddrN标志为不可用。
- Otherwise (mbAddrX is available), mbAddrN is marked as available and Table 6-4 specifies mbAddrN and yM depending on (xN, yN), currMbFrameFlag, mbIsTopMbFlag, and the following variable
- 否则（mbAddrX可用），mbAddrN标志为可用，表6-4说明依赖于(xN, yN)的mbAddrN和yM，currMbFrameFlag、mbIsTopMbFlag和下面的变量
  - If the macroblock mbAddrX is a frame macroblock, mbAddrXFrameFlag is set equal to 1,
  - 如果宏块mbAddrX是帧宏块，mbAddrXFrameFlag设置为1，
  - Otherwise (the macroblock mbAddrX is a field macroblock), mbAddrXFrameFlag is set equal to 0.
  - 否则（宏块mbAddrX是场宏块），mbAddrXFrameFlag设置为0。

Unspecified values (na) of the above flags in Table 6-4 indicate that the value of the corresponding flag is not relevant for the current table rows.

Table 6-4 - Specification of mbAddrN and yM

xN	yN	currMbFrameFlag	mbIsTopMbFlag	mbAddrX	mbAddrXFrameFlag	additional condition	mbAddrN	yM
< 0	< 0	1	1	mbAddrD			mbAddrD + 1	yN
			0	mbAddrA	1		mbAddrA	yN
					0		mbAddrA + 1	(yN + maxWH) >> 1
		0	1	mbAddrD	1		mbAddrD + 1	2*yN
					0		mbAddrD	yN
			0	mbAddrD			mbAddrD + 1	yN
< 0	0 .. maxWH - 1	1	1	mbAddrA	1		mbAddrA	yN
					0	yN % 2 = = 0	mbAddrA	yN >> 1
						yN % 2 != 0	mbAddrA + 1	yN >> 1
			0	mbAddrA	1		mbAddrA + 1	yN
					0	yN % 2 = = 0	mbAddrA	(yN + maxWH) >> 1
						yN % 2 != 0	mbAddrA + 1	(yN + maxWH) >> 1
		0	1	mbAddrA	1	yN < (maxWH / 2)	mbAddrA	yN <<1
						yN >= (maxWH / 2)	mbAddrA + 1	(yN <<1) - maxWH
					0		mbAddrA	yN
			0	mbAddrA	1	yN < (maxWH / 2)	mbAddrA	(yN <<1) + 1
						yN >= (maxWH / 2)	mbAddrA + 1	(yN <<1) + 1 – maxWH
					0		mbAddrA + 1	yN
0 .. maxWH - 1	< 0	1	1	mbAddrB			mbAddrB + 1	yN
			0	CurrMbAddr			CurrMbAddr - 1	yN
		0	1	mbAddrB	1		mbAddrB + 1	2 * yN
					0		mbAddrB	yN
			0	mbAddrB			mbAddrB + 1	yN
0 .. maxWH - 1	0 .. maxWH - 1			CurrMbAddr			CurrMbAddr	yN
> maxWH - 1	<0	1	1	mbAddrC			mbAddrC + 1	yN
			0	not available			not available	na
		0	1	mbAddrC	1		mbAddrC + 1	2 * yN
					0		mbAddrC	yN
			0	mbAddrC			mbAddrC + 1	yN
> maxWH - 1	0 .. maxWH - 1			not available			not available	na
	> maxWH - 1			not available			not available	na

The neighbouring luma location (xW, yW) relative to the upper-left corner of the macroblock mbAddrN is derived as

相对于宏块mbAddrN左上角的相邻亮度位置(xW, yW)生成如下

$$xW = (xN + maxWH) \% maxWH \qquad (6\text{-}27)$$

$$yW = (yM + maxWH) \% maxWH \qquad (6\text{-}28)$$

7 Syntax and semantics

7.1 Method of describing syntax in tabular form

7.1 以表格的形式描述语法的方法

The syntax tables describe a superset of the syntax of all allowed input bitstreams. Additional constraints on the syntax may be specified in other clauses.

语法表描述了输入码流语法的一个超集，关于语法的一些附加限制将在其他部分说明。

NOTE - An actual decoder should implement means for identifying entry points into the bitstream and to identify and handle non-conforming bitstreams. The methods for identifying and handling errors and other such situations are not described here.

注 — 一个实际的解码器应该实现识别进入码流切入点的方法，并且能够识别和处理非兼容的码流，识别和处理错误和其他类似情况的方法在这里没有描述。

The following table lists examples of pseudo code used to describe the syntax. When syntax_element appears, it specifies that a data element is read (extracted) from the bitstream and the bitstream pointer.

下表列出了用于描述语法的伪代码的一些例子。当语法元素出现的时候，说明将从码流中读入一个数据元素并且移动码流的指针。

	C	Descriptor
/* A statement can be a syntax element with an associated syntax category and descriptor or can be an expression used to specify conditions for the existence, type, and quantity of syntax elements, as in the following two examples */		
syntax_element	3	ue(v)
conditioning statement		
/* A group of statements enclosed in curly brackets is a compound statement and is treated functionally as a single statement. */		
{		
statement		
statement		
…		
}		
/* A "while" structure specifies a test of whether a condition is true, and if true, specifies evaluation of a statement (or compound statement) repeatedly until the condition is no longer true */		
while(condition)		
statement		
/* A "do … while" structure specifies evaluation of a statement once, followed by a test of whether a condition is true, and if true, specifies repeated evaluation of the statement until the condition is no longer true */		
do		
statement		
while(condition)		
/* An "if … else" structure specifies a test of whether a condition is true, and if the condition is true, specifies evaluation of a primary statement, otherwise specifies evaluation of an alternative statement. The "else" part of the structure and the associated alternative statement is omitted if no alternative statement evaluation is needed */		
if(condition)		
primary statement		
else		
alternative statement		
/* A "for" structure specifies evaluation of an initial statement, followed by a test of a condition, and if the condition is true, specifies repeated evaluation of a primary statement followed by a subsequent statement until the condition is no longer true. */		
for(initial statement; condition; subsequent statement)		
primary statement		

7.2 Specification of syntax functions, categories, and descriptors

7.2 语法函数、类别和描述符的说明

The functions presented here are used in the syntactical description. These functions assume the existence of a bitstream pointer with an indication of the position of the next bit to be read by the decoding process from the bitstream.

这里的函数被用来进行语法描述，这些函数假设存在码流指针并且根据码流的解码顺序指向下一个将要被读的比特的位置。

byte_aligned( )

- Returns TRUE if the current position in the bitstream is on a byte boundary, i.e., the next bit in the bitstream is the first bit in a byte. Otherwise it returns FALSE
- 如果在比特流中当前的位置是一个字节的边缘，例如码流中的下一个比特是一个字节的第一个比特，那么返回 TRUE， 否则返回 FALSE。

more_data_in_byte_stream( )

- Returns TRUE if more data follows in the byte stream. Otherwise it returns FALSE. Used only in the byte stream NAL unit syntax structure specified in Annex B.
- 如果在字节流中还有数据，那么返回 TRUE，否则返回 FALSE，仅仅使用在字节流 NAL 单元语法结构中，如附录 B 所示。

more_rbsp_data( )

- Returns TRUE if there is more data in an RBSP before rbsp_trailing_bits( ). Otherwise it returns FALSE. The method for enabling determination of whether there is more data in the RBSP is specified by the application (or in Annex B for applications that use the byte stream format).
- 如果在 rbsp_trailing_bits( ) 之前的 RBSP 中还有更多的数据，那么返回 TRUE，否则返回 FALSE。判断在 RBSP 中是否有更多数据的方法由具体的应用（或者在附录 B 中的使用字节流格式的应用）。来说明

more_rbsp_trailing_data( )

- Returns TRUE if there is more data in an RBSP. Otherwise it returns FALSE.
- 在 RBSP 中如果有更多的数据返回 TRUE，否则返回 FALSE。

next_bits(n)

- Provides the next bits in the bitstream for comparison purposes, without advancing the bitstream pointer. Provides a look at the next n bits in the bitstream with n being its argument. When used within the byte stream as specified in Annex B, next_bits(n) returns a value of 0 if fewer than n bits remain within the byte stream.
- 为了比较的目的，返回码流中接下来几个比特的值，但是码流的指针没有移动。如果用 n 作为参数，就返回码流中接下来 n 个比特的值。当在附录 B 中的字节流中使用时，如果在字节流中只有小于 n 个比特的数据，那么 next_bits(n) 将返回 0。

read_bits(n)

- Reads the next n bits from the bitstream and advances the bitstream pointer by n bit positions. When n is equal to 0, read_bits(n) is specified to return a value equal to 0 and to not advance the bitstream pointer.
- 读出码流中接下来 n 个比特的值并且码流指针向前移动 n 个比特位置，当 n 等于 0 时，那么 read_bits(n) 返回的值等于 0 并且码流的指针不移动。

Categories (labelled in the table as **C**) specify the partitioning of slice data into at most three slice data partitions. Slice data partition A contains all syntax elements of category 2. Slice data partition B contains all syntax elements of category 3. Slice data partition C contains all syntax elements of category 4. The meaning of other category values is not specified. For some syntax elements, two category values, separated by a vertical bar, are used. In these cases, the category value to be applied is further specified in the text. For syntax structures used within other syntax structures, the categories of all syntax elements found within the included syntax structure are listed, separated by a vertical bar. A syntax element or syntax structure with category marked as "All" is present within all syntax structures that include that syntax element or syntax structure. For syntax structures used within other syntax structures, a numeric category value provided in a syntax table at the location of the inclusion of a syntax structure containing a syntax element with category marked as "All" is considered to apply to the syntax elements with category "All".

类别（在表中标示为 C）说明 slice 数据最多可以划分成 3 种 slice 数据块，slice 数据块 A 包含了类别 2 的所有语法元素，slice 数据块 B 包含了类别 3 的所有语法元素，slice 数据块 C 包含了类别 4 的所有语法元素，其他类别值的意义在这里没有说明。对于一些语法元素，使用两种类别的值，有 | 分开。在这种情况下，具体使用哪种类别值在文中将进一步的说明。对于包括在其他语法结构中的语法结构，被包括的语法结构中的所有语法元素的类别都被列举出来，由| 分开。标有“All”的语法元素或语法结构的类别将出现在包括那个语法元素或语法结构的所有语法结构中。对于包含在其他语法结构中使用的语法结构，在语法表中包含类别标示为“All”的语法元素的语法结构位置处的类别值应用于标识为“All”的语法元素。（翻译的有点乱）

The following descriptors specify the parsing process of each syntax element. For some syntax elements, two descriptors, separated by a vertical bar, are used. In these cases, the left descriptors apply when entropy_coding_mode_flag is equal to 0 and the right descriptor applies when entropy_coding_mode_flag is equal to 1.

下面的描述说明了每一个语法元素的解析过程。对于一些语法元素，使用两个描述符，中间用 | 隔开，在这种情况下，当 entropy_coding_mode_flag 等于 0 时使用左边的描述符，当 entropy_coding_mode_flag 等于 1 时使用右边的描述符。

- ae(v): context-adaptive arithmetic entropy-coded syntax element. The parsing process for this descriptor is specified in subclause 9.3.
- ae(v)：上下文自适应算术熵编码语法元素，这个描述符的解析过程见 9.3 小节。
- b(8): byte having any pattern of bit string (8 bits). The parsing process for this descriptor is specified by the return value of the function read_bits(8).
- b(8)：任何比特串格式（8个比特）的字节，对于这个描述符的解析过程由函数 read_bits(8) 的返回值来说明。
- ce(v): context-adaptive variable-length entropy-coded syntax element with the left bit first. The parsing process for this descriptor is specified in subclause 9.2.
- ce(v)：上下文自适应的变长熵编码语法元素，首先是左边的比特，这个描述符的解析过程见 9.2 小节。
- f(n): fixed-pattern bit string using n bits written (from left to right) with the left bit first. The parsing process for this descriptor is specified by the return value of the function read_bits(n).
- f(n)：固定模式的比特串，首先写最左边比特（从左到右）的 n 个比特。这个描述符的解析过程由函数 read_bits(n) 的返回值来说明。
- i(n): signed integer using n bits. When n is "v" in the syntax table, the number of bits varies in a manner dependent on the value of other syntax elements. The parsing process for this descriptor is specified by the return value of the function read_bits(n) interpreted as a two's complement integer representation with most significant bit written first.
- i(n)：n 个比特的有符号整数，在语法表中，当 n 是“v”时，那么比特的数目是根据其它语法元素的值而变化的。这个描述符的解析过程由函数 read_bits(n) 的返回值来说明，read_bits(n) 是两个互补整数表示，首先写入最重要的比特。
- me(v): mapped Exp-Golomb-coded syntax element with the left bit first. The parsing process for this descriptor is specified in subclause 9.1.
- me(v)：用首先最左边的比特方法映射 Exp-Golomb-coded 语法元素，这个描述符的解析过程由 9.1 小节来说明。
- se(v): signed integer Exp-Golomb-coded syntax element with the left bit first. The parsing process for this descriptor is specified in subclause 9.1.
- se(v)：有符号整数 Exp-Golomb-coded 语法元素，也采用首先最左边的比特方法，这个描述符的解析过程由 9.1 小节说明。
- te(v): truncated Exp-Golomb-coded syntax element with left bit first. The parsing process for this descriptor is specified in subclause 9.1.
- te(v)：截断 Exp-Golomb-coded 语法元素，也采用首先最左边的比特方法，这个描述符的解析过程由 9.1 小节说明。
- u(n): unsigned integer using n bits. When n is "v" in the syntax table, the number of bits varies in a manner dependent on the value of other syntax elements. The parsing process for this descriptor is specified by the return value of the function read_bits(n) interpreted as a binary representation of an unsigned integer with most significant bit written first.
- u(n)：n 个比特的无符号整数，在语法表中，当 n 是“v”时，那么比特的数目是根据其它语法元素的值而变化的。这个描述符的解析过程由函数 read_bits(n) 的返回值来说明，read_bits(n) 为无符号整数的二进制表示，也是采用首先最左边的比特方法。
- ue(v): unsigned integer Exp-Golomb-coded syntax element with the left bit first. The parsing process for this descriptor is specified in subclause 9.1.

- ue(v)：无符号整数 Exp-Golomb-coded 语法元素，采用首先最左边的比特方法，这个描述符的解析过程由 9.1 小节说明。

7.3 Syntax in tabular form

7.3.1 NAL unit syntax

nal_unit(NumBytesInNALunit) {	C	Descriptor
forbidden_zero_bit	All	f(1)
nal_ref_idc	All	u(2)
nal_unit_type	All	u(5)
NumBytesInRBSP = 0		
for(i = 1; i < NumBytesInNALunit; i++) {		
if(i + 2 < NumBytesInNALunit && next_bits(24) = = 0x000003) {		
rbsp_byte[NumBytesInRBSP++ **]**	All	b(8)
rbsp_byte[NumBytesInRBSP++ **]**	All	b(8)
i += 2		
emulation_prevention_three_byte /* equal to 0x03 */	All	f(8)
} else		
rbsp_byte[NumBytesInRBSP++ **]**	All	b(8)
}		
}		

7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax

7.3.2.1 Sequence parameter set RBSP syntax

seq_parameter_set_rbsp( ) {	**C**	**Descriptor**
profile_idc	0	u(8)
constraint_set0_flag	0	u(1)
constraint_set1_flag	0	u(1)
constraint_set2_flag	0	u(1)
reserved_zero_5bits /* equal to 0 */	0	u(5)
level_idc	0	u(8)
seq_parameter_set_id	0	ue(v)
log2_max_frame_num_minus4	0	ue(v)
pic_order_cnt_type	0	ue(v)
if(pic_order_cnt_type = = 0)		
log2_max_pic_order_cnt_lsb_minus4	0	ue(v)
else if(pic_order_cnt_type = = 1) {		
delta_pic_order_always_zero_flag	0	u(1)
offset_for_non_ref_pic	0	se(v)
offset_for_top_to_bottom_field	0	se(v)
num_ref_frames_in_pic_order_cnt_cycle	0	ue(v)
for(i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++)		
offset_for_ref_frame[i **]**	0	se(v)
}		
num_ref_frames	0	ue(v)
gaps_in_frame_num_value_allowed_flag	0	u(1)
pic_width_in_mbs_minus1	0	ue(v)
pic_height_in_map_units_minus1	0	ue(v)
frame_mbs_only_flag	0	u(1)
if(!frame_mbs_only_flag)		
mb_adaptive_frame_field_flag	0	u(1)
direct_8x8_inference_flag	0	u(1)
frame_cropping_flag	0	u(1)
if(frame_cropping_flag) {		
frame_crop_left_offset	0	ue(v)
frame_crop_right_offset	0	ue(v)
frame_crop_top_offset	0	ue(v)
frame_crop_bottom_offset	0	ue(v)
}		
vui_parameters_present_flag	0	u(1)
if(vui_parameters_present_flag)		
vui_parameters( )	0	
rbsp_trailing_bits( )	0	
}		

7.3.2.2 Picture parameter set RBSP syntax

pic_parameter_set_rbsp( ) {	**C**	**Descriptor**
pic_parameter_set_id	1	ue(v)
seq_parameter_set_id	1	ue(v)
entropy_coding_mode_flag	1	u(1)
pic_order_present_flag	1	u(1)
num_slice_groups_minus1	1	ue(v)
if(num_slice_groups_minus1 > 0) {		
slice_group_map_type	1	ue(v)
if(slice_group_map_type = = 0)		
for(iGroup = 0; iGroup <= num_slice_groups_minus1; iGroup++)		
run_length_minus1[iGroup **]**	1	ue(v)
else if(slice_group_map_type = = 2)		
for(iGroup = 0; iGroup < num_slice_groups_minus1; iGroup++) {		
top_left[iGroup **]**	1	ue(v)
bottom_right[iGroup **]**	1	ue(v)
}		
else if(slice_group_map_type = = 3 || slice_group_map_type = = 4 || slice_group_map_type = = 5) {		
slice_group_change_direction_flag	1	u(1)
slice_group_change_rate_minus1	1	ue(v)
} else if(slice_group_map_type = = 6) {		
pic_size_in_map_units_minus1	1	ue(v)
for(i = 0; i <= pic_size_in_map_units_minus1; i++)		
slice_group_id[i **]**	1	u(v)
}		
}		
num_ref_idx_l0_active_minus1	1	ue(v)
num_ref_idx_l1_active_minus1	1	ue(v)
weighted_pred_flag	1	u(1)
weighted_bipred_idc	1	u(2)
pic_init_qp_minus26 /* relative to 26 */	1	se(v)
pic_init_qs_minus26 /* relative to 26 */	1	se(v)
chroma_qp_index_offset	1	se(v)
deblocking_filter_control_present_flag	1	u(1)
constrained_intra_pred_flag	1	u(1)
redundant_pic_cnt_present_flag	1	u(1)
rbsp_trailing_bits( )	1	
}		

7.3.2.3 Supplemental enhancement information RBSP syntax

sei_rbsp( ) {	**C**	**Descriptor**
do		
sei_message( )	5	
while(more_rbsp_data( ))		
rbsp_trailing_bits( )	5	
}		

7.3.2.3.1 Supplemental enhancement information message syntax

sei_message( ) {	**C**	**Descriptor**
payloadType = 0		
while(next_bits(8) = = 0xFF) {		
ff_byte /* equal to 0xFF */	5	f(8)
payloadType += 255		
}		
last_payload_type_byte	5	u(8)
payloadType += last_payload_type_byte		
payloadSize = 0		
while(next_bits(8) = = 0xFF) {		
ff_byte /* equal to 0xFF */	5	f(8)
payloadSize += 255		
}		
last_payload_size_byte	5	u(8)
payloadSize += last_payload_size_byte		
sei_payload(payloadType, payloadSize)	5	
}		

7.3.2.4 Access unit delimiter RBSP syntax

access_unit_delimiter_rbsp( ) {	**C**	**Descriptor**
primary_pic_type	6	u(3)
rbsp_trailing_bits( )	6	
}		

7.3.2.5 End of sequence RBSP syntax

end_of_seq_rbsp( ) {	**C**	**Descriptor**
}		

7.3.2.6 End of stream RBSP syntax

end_of_stream_rbsp( ) {	**C**	**Descriptor**
}		

7.3.2.7 Filler data RBSP syntax

filler_data_rbsp(NumBytesInRBSP) {	**C**	**Descriptor**
while(next_bits(8) = = 0xFF)		
ff_byte /* equal to 0xFF */	9	f(8)
rbsp_trailing_bits( )	9	
}		

7.3.2.8 Slice layer without partitioning RBSP syntax

slice_layer_without_partitioning_rbsp( ) {	**C**	**Descriptor**
slice_header( )	2	
slice_data( ) /* all categories of slice_data( ) syntax */	2 | 3 | 4	
rbsp_slice_trailing_bits( )	2	
}		

7.3.2.9 Slice data partition RBSP syntax

7.3.2.9.1 Slice data partition A RBSP syntax

slice_data_partition_a_layer_rbsp( ) {	**C**	**Descriptor**
slice_header( )	2	
slice_id	2	ue(v)
slice_data( ) /* only category 2 parts of slice_data( ) syntax */	2	
rbsp_slice_trailing_bits( )	2	
}		

7.3.2.9.2 Slice data partition B RBSP syntax

slice_data_partition_b_layer_rbsp( ) {	**C**	**Descriptor**
slice_id	3	ue(v)
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	3	ue(v)
slice_data( ) /* only category 3 parts of slice_data( ) syntax */	3	
rbsp_slice_trailing_bits( )	3	
}		

7.3.2.9.3 Slice data partition C RBSP syntax

slice_data_partition_c_layer_rbsp( ) {	C	Descriptor
slice_id	4	ue(v)
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	4	ue(v)
slice_data( ) /* only category 4 parts of slice_data( ) syntax */	4	
rbsp_slice_trailing_bits( )	4	
}		

7.3.2.10 RBSP slice trailing bits syntax

rbsp_slice_trailing_bits( ) {	C	Descriptor
rbsp_trailing_bits( )	All	
if(entropy_coding_mode_flag)		
while(more_rbsp_trailing_data( ))		
cabac_zero_word /* equal to 0x0000 */	All	f(16)
}		

7.3.2.11 RBSP trailing bits syntax

rbsp_trailing_bits( ) {	C	Descriptor
rbsp_stop_one_bit /* equal to 1 */	All	f(1)
while(!byte_aligned( ))		
rbsp_alignment_zero_bit /* equal to 0 */	All	f(1)
}		

7.3.3 Slice header syntax

slice_header( ) {	C	Descriptor
first_mb_in_slice	2	ue(v)
slice_type	2	ue(v)
pic_parameter_set_id	2	ue(v)
frame_num	2	u(v)
if(!frame_mbs_only_flag) {		
field_pic_flag	2	u(1)
if(field_pic_flag)		
bottom_field_flag	2	u(1)
}		
if(nal_unit_type = = 5)		
idr_pic_id	2	ue(v)
if(pic_order_cnt_type = = 0) {		
pic_order_cnt_lsb	2	u(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt_bottom	2	se(v)
}		
if(pic_order_cnt_type = = 1 && !delta_pic_order_always_zero_flag) {		
delta_pic_order_cnt[0 **]**	2	se(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt[1 **]**	2	se(v)
}		
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	2	ue(v)
if(slice_type = = B)		
direct_spatial_mv_pred_flag	2	u(1)
if(slice_type = = P | | slice_type = = SP | | slice_type = = B) {		
num_ref_idx_active_override_flag	2	u(1)
if(num_ref_idx_active_override_flag) {		
num_ref_idx_l0_active_minus1	2	ue(v)
if(slice_type = = B)		
num_ref_idx_l1_active_minus1	2	ue(v)
}		
}		
ref_pic_list_reordering( )	2	
if((weighted_pred_flag && (slice_type = = P || slice_type = = SP)) || (weighted_bipred_idc = = 1 && slice_type = = B))		
pred_weight_table( )	2	
if(nal_ref_idc != 0)		
dec_ref_pic_marking( )	2	
if(entropy_coding_mode_flag && slice_type != I && slice_type != SI)		
cabac_init_idc	2	ue(v)
slice_qp_delta	2	se(v)
if(slice_type = = SP || slice_type = = SI) {		
if(slice_type = = SP)		
sp_for_switch_flag	2	u(1)
slice_qs_delta	2	se(v)

}		
if(deblocking_filter_control_present_flag) {		
disable_deblocking_filter_idc	2	ue(v)
if(disable_deblocking_filter_idc != 1) {		
slice_alpha_c0_offset_div2	2	se(v)
slice_beta_offset_div2	2	se(v)
}		
}		
if(num_slice_groups_minus1 > 0 && slice_group_map_type >= 3 && slice_group_map_type <= 5)		
slice_group_change_cycle	2	u(v)
}		

7.3.3.1 Reference picture list reordering syntax

ref_pic_list_reordering( ) {	**C**	**Descriptor**
if(slice_type != I && slice_type != SI) {		
ref_pic_list_reordering_flag_l0	2	u(1)
if(ref_pic_list_reordering_flag_l0)		
do {		
reordering_of_pic_nums_idc	2	ue(v)
if(reordering_of_pic_nums_idc = = 0 || reordering_of_pic_nums_idc = = 1)		
abs_diff_pic_num_minus1	2	ue(v)
else if(reordering_of_pic_nums_idc = = 2)		
long_term_pic_num	2	ue(v)
} while(reordering_of_pic_nums_idc != 3)		
}		
if(slice_type = = B) {		
ref_pic_list_reordering_flag_l1	2	u(1)
if(ref_pic_list_reordering_flag_l1)		
do {		
reordering_of_pic_nums_idc	2	ue(v)
if(reordering_of_pic_nums_idc = = 0 || reordering_of_pic_nums_idc = = 1)		
abs_diff_pic_num_minus1	2	ue(v)
else if(reordering_of_pic_nums_idc = = 2)		
long_term_pic_num	2	ue(v)
} while(reordering_of_pic_nums_idc != 3)		
}		
}		

7.3.3.2 Prediction weight table syntax

pred_weight_table( ) {	**C**	**Descriptor**
luma_log2_weight_denom	2	ue(v)
chroma_log2_weight_denom	2	ue(v)
for(i = 0; i <= num_ref_idx_l0_active_minus1; i++) {		
luma_weight_l0_flag	2	u(1)
if(luma_weight_l0_flag) {		
luma_weight_l0[i **]**	2	se(v)
luma_offset_l0[i **]**	2	se(v)
}		
chroma_weight_l0_flag	2	u(1)
if(chroma_weight_l0_flag)		
for(j =0; j < 2; j++) {		
chroma_weight_l0[i **][** j **]**	2	se(v)
chroma_offset_l0[i **][** j **]**	2	se(v)
}		
}		
if(slice_type = = B)		
for(i = 0; i <= num_ref_idx_l1_active_minus1; i++) {		
luma_weight_l1_flag	2	u(1)
if(luma_weight_l1_flag) {		
luma_weight_l1[i **]**	2	se(v)
luma_offset_l1[i **]**	2	se(v)
}		
chroma_weight_l1_flag	2	u(1)
if(chroma_weight_l1_flag)		
for(j = 0; j < 2; j++) {		
chroma_weight_l1[i **][** j **]**	2	se(v)
chroma_offset_l1[i **][** j **]**	2	se(v)
}		
}		
}		

7.3.3.3 Decoded reference picture marking syntax

dec_ref_pic_marking( ) {	**C**	**Descriptor**
if(nal_unit_type == 5) {		
no_output_of_prior_pics_flag	2 | 5	u(1)
long_term_reference_flag	2 | 5	u(1)
} else {		
adaptive_ref_pic_marking_mode_flag	2 | 5	u(1)
if(adaptive_ref_pic_marking_mode_flag)		
do {		
memory_management_control_operation	2 | 5	ue(v)
if(memory_management_control_operation == 1 || memory_management_control_operation == 3)		
difference_of_pic_nums_minus1	2 | 5	ue(v)
if(memory_management_control_operation == 2)		
long_term_pic_num	2 | 5	ue(v)
if(memory_management_control_operation == 3 || memory_management_control_operation == 6)		
long_term_frame_idx	2 | 5	ue(v)
if(memory_management_control_operation == 4)		
max_long_term_frame_idx_plus1	2 | 5	ue(v)
} while(memory_management_control_operation != 0)		
}		
}		

7.3.4 Slice data syntax

slice_data( ) {	**C**	**Descriptor**
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
mb_skip_flag	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 = = 0 || (CurrMbAddr % 2 = = 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && CurrMbAddr % 2 = = 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

7.3.5 Macroblock layer syntax

macroblock_layer( ) {	**C**	**Descriptor**
mb_type	2	ue(v) | ae(v)
if(mb_type = = I_PCM) {		
while(!byte_aligned())		
pcm_alignment_zero_bit	2	f(1)
for(i = 0; i < 256 * ChromaFormatFactor; i++)		
pcm_byte[i **]**	2	u(8)
} else {		
if(MbPartPredMode(mb_type, 0) != Intra_4x4 && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) = = 4)		
sub_mb_pred(mb_type)	2	
else		
mb_pred(mb_type)	2	
if(MbPartPredMode(mb_type, 0) != Intra_16x16)		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual( )	3 | 4	
}		
}		
}		

7.3.5.1 Macroblock prediction syntax

mb_pred(mb_type) {	**C**	**Descriptor**
if(MbPartPredMode(mb_type, 0) = = Intra_4x4 || MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) = = Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

7.3.5.2 Sub-macroblock prediction syntax

sub_mb_pred(mb_type) {	**C**	**Descriptor**
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		

if((num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) &&
      mb_type != P_8x8ref0 &&
      sub_mb_type[mbPartIdx] != B_Direct_8x8 &&
      SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		

ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		

if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) &&
      sub_mb_type[mbPartIdx] != B_Direct_8x8 &&
      SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		

ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		

for(subMbPartIdx = 0;
          subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]);
          subMbPartIdx++)		

for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		

for(subMbPartIdx = 0;
          subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]);
          subMbPartIdx++)		

for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

7.3.5.3 Residual data syntax

residual( ) {	**C**	**Descriptor**
if(!entropy_coding_mode_flag)		
residual_block = residual_block_cavlc		
else		
residual_block = residual_block_cabac		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
residual_block(Intra16x16DCLevel, 16)	3	
for(i8x8 = 0; i8x8 < 4; i8x8++) /* each luma 8x8 block */		
for(i4x4 = 0; i4x4 < 4; i4x4++) /* each 4x4 sub-block of block */		
if(CodedBlockPatternLuma & (1 << i8x8)) {		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
residual_block(Intra16x16ACLevel[i8x8 * 4 + i4x4], 15)	3	
else		
residual_block(LumaLevel[i8x8 * 4 + i4x4], 16)	3 | 4	
} else {		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
for(i = 0; i < 15; i++)		
Intra16x16ACLevel[i8x8 * 4 + i4x4][i] = 0		
else		
for(i = 0; i < 16; i++)		
LumaLevel[i8x8 * 4 + i4x4][i] = 0		
}		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
if(CodedBlockPatternChroma & 3) /* chroma DC residual present */		
residual_block(ChromaDCLevel[iCbCr], 4)	3 | 4	
else		
for(i = 0; i < 4; i++)		
ChromaDCLevel[iCbCr][i] = 0		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
for(i4x4 = 0; i4x4 < 4; i4x4++)		
if(CodedBlockPatternChroma & 2) /* chroma AC residual present */		
residual_block(ChromaACLevel[iCbCr][i4x4], 15)	3 | 4	
else		
for(i = 0; i < 15; i++)		
ChromaACLevel[iCbCr][i4x4][i] = 0		
}		

7.3.5.3.1 Residual block CAVLC syntax

residual_block_cavlc(coeffLevel, maxNumCoeff) {	**C**	**Descriptor**
for(i = 0; i < maxNumCoeff; i++)		
coeffLevel[i] = 0		
coeff_token	3 | 4	ce(v)
if(TotalCoeff(coeff_token) > 0) {		
if(TotalCoeff(coeff_token) > 10 && TrailingOnes(coeff_token) < 3)		
suffixLength = 1		
else		
suffixLength = 0		
for(i = 0; i < TotalCoeff(coeff_token); i++)		
if(i < TrailingOnes(coeff_token)) {		
trailing_ones_sign_flag	3 | 4	u(1)
level[i] = 1 – 2 * trailing_ones_sign_flag		
} else {		
level_prefix	3 | 4	ce(v)
levelCode = (level_prefix << suffixLength)		
if(suffixLength > 0 || level_prefix >= 14) {		
level_suffix	3 | 4	u(v)
levelCode += level_suffix		
}		
if(level_prefix = = 14 && suffixLength = = 0)		
levelCode += 15		
if(i = = TrailingOnes(coeff_token) && TrailingOnes(coeff_token) < 3)		
levelCode += 2		
if(levelCode % 2 = = 0)		
level[i] = (levelCode + 2) >> 1		
else		
level[i] = (–levelCode – 1) >> 1		
if(suffixLength = = 0)		
suffixLength = 1		
if(Abs(level[i]) > (3 << (suffixLength – 1)) && suffixLength < 6)		
suffixLength++		
}		
if(TotalCoeff(coeff_token) < maxNumCoeff) {		
total_zeros	3 | 4	ce(v)
zerosLeft = total_zeros		
} else		
zerosLeft = 0		
for(i = 0; i < TotalCoeff(coeff_token) – 1; i++) {		
if(zerosLeft > 0) {		
run_before	3 | 4	ce(v)
run[i] = run_before		
} else		
run[i] = 0		
zerosLeft = zerosLeft – run[i]		
}		

run[TotalCoeff(coeff_token) – 1] = zerosLeft		
coeffNum = -1		
for(i = TotalCoeff(coeff_token) – 1; i >= 0; i--) {		
coeffNum += run[i] + 1		
coeffLevel[coeffNum] = level[i]		
}		
}		
}		

7.3.5.3.2 Residual block CABAC syntax

residual_block_cabac(coeffLevel, maxNumCoeff) {	**C**	**Descriptor**
coded_block_flag	3 | 4	ae(v)
if(coded_block_flag) {		
numCoeff = maxNumCoeff		
i = 0		
do {		
significant_coeff_flag[i **]**	3 | 4	ae(v)
if(significant_coeff_flag[i]) {		
last_significant_coeff_flag[i **]**	3 | 4	ae(v)
if(last_significant_coeff_flag[i]) {		
numCoeff = i + 1		
for(j = numCoeff; j < maxNumCoeff; j++)		
coeffLevel[j] = 0		
}		
}		
i++		
} while(i < numCoeff-1)		
coeff_abs_level_minus1[numCoeff-1 **]**	3 | 4	ae(v)
coeff_sign_flag[numCoeff-1 **]**	3 | 4	ae(v)
coeffLevel[numCoeff-1] = (coeff_abs_level_minus1[numCoeff – 1] + 1) * (1 – 2 * coeff_sign_flag[numCoeff – 1])		
for(i = numCoeff-2; i >= 0; i--) {		
if(significant_coeff_flag[i]) {		
coeff_abs_level_minus1[i **]**	3 | 4	ae(v)
coeff_sign_flag[i **]**	3 | 4	ae(v)
coeffLevel[i] = (coeff_abs_level_minus1[i] + 1) * (1 – 2 * coeff_sign_flag[i])		
} else		
coeffLevel[i] = 0		
}		
} else		
for(i = 0; i < maxNumCoeff; i++)		
coeffLevel[i] = 0		
}		

7.4 Semantics

7.4 语义

7.4.1 NAL unit semantics

7.4.1 NAL 单元语义

NOTE - The VCL is specified to efficiently represent the content of the video data. The NAL is specified to format that data and provide header information in a manner appropriate for conveyance on a variety of communication channels or storage media. All data are contained in NAL units, each of which contains an integer number of bytes. A NAL unit specifies a generic format for use in both packet-oriented and bitstream systems. The format of NAL units for both packet-oriented transport and byte stream is identical except that each NAL unit can be preceded by a start code prefix and extra padding bytes in the byte stream format.

注 － VCL 被用来表示视频数据的内容，NAL 则用来格式化那些数据并且用适当的方式来加一些头信息，以适应各种不同的通信链路和存储媒体。NAL 单元包含了所有的数据，每一个 NAL 单元包含有整数个字节。NAL单元说明了在面向包和比特流系统中的一般格式。对于面向包的传输和字节流来说，NAL 单元的格式是相同的，除了在字节流格式中每个NAL单元的前面有前缀开始码和一些特殊的填充字节外。

NumBytesInNALunit specifies the size of the NAL unit in bytes. This value is required for decoding of the NAL unit. Some form of demarcation of NAL unit boundaries is necessary to enable inference of NumBytesInNALunit. One such demarcation method is specified in Annex B for the byte stream format. Other methods of demarcation may be specified outside of this Recommendation | International Standard.

NumBytesInNALunit 说明了NAL 单元的大小（以字节为单位），在解码 NAL 单元时需要使用这个值。一些划分 NAL 单元边界的方法必须给 NumBytesInNALunit 变量赋值。对于字节流格式的划分方法在附录 B 中说明，其他的划分方法属于本协议之外的内容。

forbidden_zero_bit shall be equal to 0.

forbidden_zero_bit 应该等于 0。

nal_ref_idc not equal to 0 specifies that the content of the NAL unit contains a sequence parameter set or a picture parameter set or a slice of a reference picture or a slice data partition of a reference picture.

nal_ref_idc equal to 0 for a NAL unit containing a slice or slice data partition indicates that the slice or slice data partition is part of a non-reference picture.

nal_ref_idc shall not be equal to 0 for sequence parameter set or picture parameter set NAL units. When nal_ref_idc is equal to 0 for one slice or slice data partition NAL unit of a particular picture, it shall be equal to 0 for all slice and slice data partition NAL units of the picture.

nal_ref_idc shall be not be equal to 0 for IDR NAL units, i.e., NAL units with nal_unit_type equal to 5.

nal_ref_idc shall be equal to 0 for all NAL units having nal_unit_type equal to 6, 9, 10, 11, or 12.

nal_ref_idc 不等于 0 说明 NAL 单元中的内容是序列参数集或者图像参数集或者参考图像的 slice 或者参考图像的slice 数据块。

nal_ref_idc 等于 0 说明 NAL 单元中的 slice 或者 slice 数据块是非参考图像的 slice 或者 slice 数据块。

对于 序列参数集和图像参数集的 NAL 单元，nal_ref_idc 不等于 0。对于图像的一个 slice 或者 slice 数据块的 NAL 单元来说，如果 nal_ref_idc 等于 0，那么对于那个图像的所有 slice 或者 slice 数据块的 NAL 单元来说，nal_ref_idc 都等于 0。

对于 IDR NAL 单元，nal_ref_idc 不等于 0，例如：NAL 单元的 nal_unit_type 等于 5。

对于 nal_unit_type 等于 6, 9, 10, 11, or 12 的NAL 单元，nal_ref_idc 等于 0。

nal_unit_type specifies the type of RBSP data structure contained in the NAL unit as specified in Table 7-1. VCL NAL units are specified as those NAL units having nal_unit_type equal to 1 to 5, inclusive. All remaining NAL units are called non-VCL NAL units.

nal_unit_type 说明了包含在 NAL 单元中的 RBSP 数据结构的类型，如 Table 7-1 所示。VCL NAL 单元表示的是那些 nal_unit_type 等于 1 到 5 的 NAL 单元，包括 1 和 5，所有剩余的 NAL 单元都被称为 non-VCL NAL 单元。

The column marked "C" in Table 7-1 lists the categories of the syntax elements that may be present in the NAL unit. In addition, syntax elements with syntax category "All" may be present, as determined by the syntax and semantics of the RBSP data structure. The presence or absence of any syntax elements of a particular listed category is determined from the syntax and semantics of the associated RBSP data structure. nal_unit_type shall not be equal to 3 or 4 unless at least one syntax element is present in the RBSP data structure having a syntax element category value equal to the value of nal_unit_type and not categorized as "All".

在 Table 7-1 中标识为 “C”的列列出了在 NAL 单元中可能出现的语法元素的类别。另外，根据 RBSP 数据结构的语法和语义，也可能出现语法类别为“All”的语法元素。一个具体类别的任何一个语法元素是否出现都是由对应的 RBSP 数据结构的语法和语义来决定的。nal_unit_type 不等于 3 或者 4，除非在 RBSP 数据结构中至少出现了一个语法元素的类别值等于 nal_unit_type 值并且不是标识为“All”的语法元素。（最后一句的翻译）

Table 7-1 – NAL unit type codes

nal_unit_type	Content of NAL unit and RBSP syntax structure	C
0	Unspecified	
1	Coded slice of a non-IDR picture slice_layer_without_partitioning_rbsp( )	2, 3, 4
2	Coded slice data partition A slice_data_partition_a_layer_rbsp( )	2
3	Coded slice data partition B slice_data_partition_b_layer_rbsp( )	3
4	Coded slice data partition C slice_data_partition_c_layer_rbsp( )	4
5	Coded slice of an IDR picture slice_layer_without_partitioning_rbsp( )	2, 3
6	Supplemental enhancement information (SEI) sei_rbsp( )	5
7	Sequence parameter set seq_parameter_set_rbsp( )	0
8	Picture parameter set pic_parameter_set_rbsp( )	1
9	Access unit delimiter access_unit_delimiter_rbsp( )	6
10	End of sequence end_of_seq_rbsp( )	7
11	End of stream end_of_stream_rbsp( )	8
12	Filler data filler_data_rbsp( )	9
13..23	Reserved	
24..31	Unspecified	

In the text, coded slice NAL unit collectively refers to a coded slice of a non-IDR picture NAL unit or to a coded slice of an IDR picture NAL unit.

No decoding process for nal_unit_type equal to 0 or in the range of 24 to 31, inclusive, is specified in this Recommendation | International Standard.

NOTE – NAL unit types 0 and 24..31 may be used as determined by the application.

Decoders shall ignore (remove from the bitstream and discard) the contents of all NAL units that use reserved values of nal_unit_type.

When the value of nal_unit_type is equal to 5 for a NAL unit containing a slice of a coded picture, the value of nal_unit_type shall be 5 in all other VCL NAL units of the same coded picture. Such a picture is referred to as an IDR picture.

NOTE – Slice data partitioning cannot be used for IDR pictures.

在文中，编码 slice NAL 单元指的是非 IDR 图像的编码 slice NAL 单元或者 IDR 图像的编码 slice NAL 单元。

在本协议中没有说明 nal_unit_type 等于 0 或者 等于 24 到 31（包括 24 和 31）的 NAL 单元的解码过程。

注 — nal_unit_type 等于 0 或者 等于 24 到 31（包括 24 和 31）的 NAL 单元的使用由具体的应用来决定。

解码器将忽略 nal_unit_type 等于保留值的所有 NAL 单元的内容（从码流中删除或者忽略）。

对于一个包含编码图像 slice 的 NAL 单元来说，如果其 nal_unit_type 等于 5，那么对于相同编码图像的所有VCL NAL 单元来说，nal_unit_type 的值都等于 5，这样的图像也被称为 IDR 图像。

注 — 对于 IDR 图像来说，不能使用 数据分割方法。（？？？）

rbsp_byte[i **]** is the i-th byte of an RBSP. An RBSP is specified as an ordered sequence of bytes as follows.

rbsp_byte[i **]** 是 RBSP 中的第 i 个字节，一个 RBSP 是有序的字节序列，如下所示。

The RBSP contains an SODB in the following form:

- If the SODB is empty (i.e., zero bits in length), the RBSP is also empty.
- Otherwise, the RBSP contains the SODB in the following form:
  1) The first byte of the RBSP contains the (most significant, left-most) eight bits of the SODB; the next byte of the RBSP shall contain the next eight bits of the SODB, etc., until fewer than eight bits of the SODB remain.
  2) rbsp_trailing_bits( ) are present after the SODB as follows:
     i) The first (most significant, left-most) bits of the final RBSP byte contains the remaining bits of the SODB, (if any)
     ii) The next bit consists of a single rbsp_stop_one_bit equal to 1, and
     iii) When the rbsp_stop_one_bit is not the last bit of a byte-aligned byte, one or more rbsp_alignment_zero_bit is present to result in byte alignment.
  3) One or more cabac_zero_word 16-bit syntax elements equal to 0x0000 may be present in some RBSPs after the rbsp_trailing_bits( ) at the end of the RBSP.

RBSP 以下面的形式包含 SODB：

- 如果 SODB 是空的（例如：0 比特长度），那么 RBSP 也是空的；
- 否则，RBSP 以下面的形式包含 SODB：
  1）RBSP的第一个字节包含了 SODB 的 8 个比特（最重要的在最左边）， RBSP中接下来的字节包含了 SODB 接下来的 8 个比特，如此继续，一直到 SODB中的比特数少于 8。
  2）在 SODB 之后将是 rbsp_trailing_bits( )，如下所示：
     i) 最后的 RBSP字节中的前几个比特是 SODB中剩余的比特（如果有的话）；
     ii) 接下来的比特是等于 1 的 rbsp_stop_one_bit；
     iii) 当 rbsp_stop_one_bit 不是对齐字节中的最后一个比特时，那么将填充一个或多个 rbsp_alignment_zero_bit 进行字节对齐；
  3）在一些 RBSP 中，在 RBSP 结尾处和 rbsp_trailing_bits( ) 之后可能出现一个或者多个 16 位等于 0x0000 的语法元素 cabac_zero_word。

Syntax structures having these RBSP properties are denoted in the syntax tables using an "_rbsp" suffix. These structures shall be carried within NAL units as the content of the rbsp_byte[i] data bytes. The association of the RBSP syntax structures to the NAL units shall be as specified in Table 7-1.

具有这些 RBSP 性质的语法结构在语法表中将用 “_rbsp”后缀进行标识。这些结构将以 rbsp_byte[i] 数据字节的形式包含在 NAL 单元中。RBSP 的语法结构与 NAL 单元的关系由表 Table 7-1 说明。

NOTE - When the boundaries of the RBSP are known, the decoder can extract the SODB from the RBSP by concatenating the bits of the bytes of the RBSP and discarding the rbsp_stop_one_bit, which is the last (least significant, right-most) bit equal to 1, and discarding any following (less significant, farther to the right) bits that follow it, which are equal to 0. The data necessary for the decoding process is contained in the SODB part of the RBSP.

注 — 当知道 RBSP的边界之后，解码器将通过连接 RBSP 字节和去除 rbsp_stop_one_bit的方法从 RBSP 中提取出 SODB， rbsp_stop_one_bit 是最后一个等于 1 的比特，并且还要去除 rbsp_stop_one_bit 之后的所有比特，这些比特都等于 0。解码过程所需要的数据都包括在 RBSP中的 SODB 部分。

emulation_prevention_three_byte is a byte equal to 0x03. When an emulation_prevention_three_byte is present in the NAL unit, it shall be discarded.

emulation_prevention_three_byte 是等于 0x03 的字节，当在码流中出现 emulation_prevention_three_byte 时，它将被忽略。

The last byte of the NAL unit shall not be equal to 0x00.

NAL 单元的最后一个字节不等于 0x00。

Within the NAL unit, the following three-byte sequences shall not occur at any byte-aligned position:

在 NAL 单元中，在字节对齐的位置不可能出现下面 3 字节序列：

– 0x000000
– 0x000001
– 0x000002

Within the NAL unit, any four-byte sequence that starts with 0x000003 other than the following sequences shall not occur at any byte-aligned position:

在 NAL 单元中，在字节对齐的位置不可能出现以 0x000003 开头的而且不是下面列出的四字节序列。

– 0x00000300
– 0x00000301
– 0x00000302
– 0x00000303

7.4.1.1 Encapsulation of an SODB within an RBSP (informative)

7.4.1.1 在 RBSP 中 SODB 的封装

This subclause does not form an integral part of this Recommendation | International Standard.

本小节不是协议的必要部分。

The form of encapsulation of an SODB within an RBSP and the use of the emulation_prevention_three_byte for encapsulation of an RBSP within a NAL unit is specified for the following purposes:

为了下面的目的，在 RBSP 中封装 SODB 的形式和在 NAL 单元中为了封装 RBSP 使用 emulation_prevention_three_byte 的说明如下：

– to prevent the emulation of start codes within NAL units while allowing any arbitrary SODB to be represented within a NAL unit,
– to enable identification of the end of the SODB within the NAL unit by searching the RBSP for the rbsp_stop_one_bit starting at the end of the RBSP, and
– to enable a NAL unit to have a size larger than that of the SODB under some circumstances (using one or more cabac_zero_word).
– 在 NAL 单元中表示任意的 SODB时，阻止 NAL 单元中开始码的混淆；
– 在 NAL 单元中，通过搜索在 RBSP 结尾处的 rbsp_stop_one_bit 来辨别 SODB的结束；
– 在一些情况下（使用一个或者多个 cabac_zero_word）使 NAL 单元大于 SODB；

The encoder can produce a NAL unit from an RBSP by the following procedure:

The RBSP data is searched for byte-aligned bits of the following binary patterns:

'00000000 00000000 000000xx' (where xx represents any 2 bit pattern: 00, 01, 10, or 11),

and a byte equal to 0x03 is inserted to replace these bit patterns with the patterns

'00000000 00000000 00000011 000000xx',

and finally, when the last byte of the RBSP data is equal to 0x00 (which can only occur when the RBSP ends in a cabac_zero_word), a final byte equal to 0x03 is appended to the end of the data.

编码器通过下面的顺序从 RBSP 产生一个 NAL 单元：

在下面二进制格式的字节对齐比特位置处搜索 RBSP 数据：

'00000000 00000000 000000xx'（这里 xx 表示任何 2 比特模式：00，01，10 或者 11），

并且通过插入一个等于 0x03 的字节来取代这些比特模式，如下所示：

'00000000 00000000 00000011 000000xx'，

最后，当 RBSP 数据的最后一个字节等于 0x00 （当 RBSP 以 cabac_zero_word 结束才会发生）时，那么在数据的结尾处将附加一个等于 0x03 的字节。

The resulting sequence of bytes is then prefixed with the first byte of the NAL unit containing the indication of the type of RBSP data structure it contains. This results in the construction of the entire NAL unit.

在上述的字节序列前再前缀一个 NAL 单元的第一个字节，该字节表示了包含的 RBSP 数据结构的类型，这就完成了整个 NAL 单元的构造。

This process can allow any SODB to be represented in a NAL unit while ensuring that

这个过程允许将任何 SODB 表示成 NAL 单元，而且保证：

– no byte-aligned start code prefix is emulated within the NAL unit, and
– no sequence of 8 zero-valued bits followed by a start code prefix, regardless of byte-alignment, is emulated within the NAL unit.
– 在 NAL 单元中，没有和字节对齐的前缀开始码混淆的码子；
– 在 NAL 单元中，不管是否字节对齐，在前缀开始码后，都没有 8 个 0 值比特序列。

7.4.1.2 Order of NAL units and association to coded pictures, access units, and video sequences

7.4.1.2 NAL 单元的顺序和与编码图像、访问单元、视频序列的关系

This subclause specifies constraints on the order of NAL units in the bitstream. Any order of NAL units in the bitstream obeying these constraints is referred to in the text as the decoding order of NAL units. Within a NAL unit, the syntax in subclauses 7.3, D.1, and E.1 specifies the decoding order of syntax elements. Decoders conforming to this Recommendation | International Standard shall be capable of receiving NAL units and their syntax elements in decoding order.

本小节说明了对码流中 NAL 单元顺序的限制。码流中遵循这些限制的 NAL 单元的任何顺序都被称为 NAL 单元的解码顺序。在 NAL 单元内，7.3, D.1, 和 E.1 小节中的语法说明了语法元素的解码顺序。遵循本协议的解码器应该能够按解码顺序来接收 NAL 单元和它们的语法元素。

7.4.1.2.1 Order of sequence and picture parameter set RBSPs and their activation

7.4.1.2.1 序列和图像参数集 RBSPs 的顺序和它们的启用

NOTE – The sequence and picture parameter set mechanism decouples the transmission of infrequently changing information from the transmission of coded macroblock data. Sequence and picture parameter sets may, in some applications, be conveyed "out-of-band" using a reliable transport mechanism.

注 – 序列和图像参数集机制降低了和编码宏块数据传输相比不经常改变的信息的传输。在一些应用中，序列和图像参数集也许使用可靠的传输机制进行“out-of-band”传输。

A picture parameter set RBSP includes parameters that can be referred to by the coded slice NAL units or coded slice data partition A NAL units of one or more coded pictures.

图像参数集 RBSP 中包括一个或多个编码图像的编码 slice NAL 单元或者编码 slice 数据块 A 的 NAL 单元参考的参数。

When a picture parameter set RBSP (with a particular value of pic_parameter_set_id) is first referred to by a coded slice NAL unit or coded slice data partition A NAL unit (using that value of pic_parameter_set_id), it is activated. This picture parameter set RBSP is called the active picture parameter set RBSP until another picture parameter set RBSP is activated. A picture parameter set RBSP shall be available to the decoding process prior to its activation.

当一个图像参数集 RBSP（有 pic_parameter_set_id 来表示）被一个编码 slice NAL 单元或者编码 slice 数据块 A 的 NAL 单元参考时（使用 pic_parameter_set_id），该参数集将被激活。这个图像参数集 RBSP 被称为有效图像参数集 RBSP，直到另一个图像参数集 RBSP 被激活时为止。在它激活之前对于解码过程来说图像参数集 RBSP 是可用的。

Any picture parameter set NAL unit containing the value of pic_parameter_set_id for an active picture parameter set RBSP shall have the same content as that of the active picture parameter set RBSP unless it follows the last VCL NAL unit of a coded picture and precedes the first VCL NAL unit of another coded picture.

对于一个有效的图像参数集 RBSP 来说，包含 pic_parameter_set_id 值的图像参数集 NAL 单元和有效的图像参数集 RBSP 有相同的内容，除了后面一直到另外一编码图像第一个 VCL NAL 单元之前的编码图像的 VCL NAL 单元。

A sequence parameter set RBSP includes parameters that can be referred to by one or more picture parameter set RBSPs or one or more SEI NAL units containing a buffering period SEI message.

序列参数集 RBSP 中包含一个或多个图像参数集 RBSP 参考的参数集，或者包含缓冲周期 SEI 消息的 SEI NAL 单元参考的参数集。

When a sequence parameter set RBSP (with a particular value of seq_parameter_set_id) is first referred to by activation of a picture parameter set RBSP (using that value of seq_parameter_set_id) or is first referred to by an SEI NAL unit containing a buffering period SEI message (using that value of seq_parameter_set_id), it is activated. This sequence parameter set RBSP is called the active sequence parameter set RBSP until another sequence parameter set RBSP is activated. A sequence parameter set RBSP shall be available to the decoding process prior to its activation. An activated sequence parameter set RBSP shall remain active for the entire coded video sequence.

当一个序列参数集（由 seq_parameter_set_id 来标识）被图像参数集或者被包含缓冲期 SEI 消息的 SEI NAL 单元参考时（使用 seq_parameter_set_id），该参数集被激活。这个序列参数集 RBSP 被称为有效参数集，一直到另外一个序列参数集 RBSP 被激活时为止。在它激活之前对于解码过程来说序列参数集 RBSP 是可用的。

Any sequence parameter set NAL unit containing the value of seq_parameter_set_id for an active sequence parameter set RBSP shall have the same content as that of the active sequence parameter set RBSP unless it follows the last access unit of a coded video sequence and precedes the first VCL NAL unit and the first SEI NAL unit containing a buffering period SEI message (when present) of another coded video sequence.

对于一个有效序列参数集 RBSP 来说，包含 seq_parameter_set_id 值的序列参数集 NAL 单元和有效序列参数集 RBSP 有相同的内容，除了后面一直到另外一编码视频序列的第一个 VCL NAL 单元和第一个包含缓冲周期 SEI 消息的 SEI NAL 单元之前的编码视频序列的访问单元。

NOTE – If picture parameter set RBSP or sequence parameter set RBSP are conveyed within the bitstream, these constraints impose an order constraint on the NAL units that contain the picture parameter set RBSP or sequence parameter set RBSP, respectively. Otherwise (picture parameter set RBSP or sequence parameter set RBSP are conveyed by other means not specified in this Recommendation | International Standard), they shall be available to the decoding process in a timely fashion such that these constraints are obeyed.

注 – 如果图像参数集或者序列参数集 RBSP 在码流中进行传输，那么对包含图像参数集或者序列参数集 RBSP 的NAL 单元顺序必须使用这些限制。否则（通过其他方式传输图像参数集或者序列参数集 RBSP的话，本协议没有做明确的说明），只要遵循这些限制，对于解码过程它们也是可用的。

During operation of the decoding process (see clause 8), the values of parameters of the active picture parameter set and the active sequence parameter set shall be considered in effect. For interpretation of SEI messages, the values of the parameters of the picture parameter set and sequence parameter set that are active for the operation of the decoding process for the VCL NAL units of the primary coded picture in the same access unit shall be considered in effect unless otherwise specified in the SEI message semantics.

在解码的过程中（见第 8 部分），有效图像参数集和有效序列参数集的参数值都被认为是有效的。对于 SEI 消息的解释，在相同访问单元（访问单元的定义见下面）中的基本编码图像的 VCL NAL 单元解码操作中，图像参数集和序列参数集的参数值将被认为有效，除非在 SEI 消息语法中明确的说明。

7.4.1.2.2 Order of access units and association to coded video sequences

7.4.1.2.2 访问单元的顺序和与编码视频序列的关系

A bitstream conforming to this Recommendation | International Standard consists of one or more coded video sequences.

遵循本协议的码流由一个或多个编码视频序列组成

A coded video sequence consists of one or more access units. The order of NAL units and coded pictures and their association to access units is described in subclause 7.4.1.2.3.

一编码视频序列由一个或者多个访问单元组成。NAL 单元的顺序和编码图像和它们与访问单元的关系在 7.4.1.2.3 小节中进行描述。

The first access unit of each coded video sequence is an IDR access unit. All subsequent access units in the coded video sequence are non-IDR access units.

每个编码视频序列的第一个访问单元是 IDR 访问单元，视频序列中接下来所有的访问单元都是 non-IDR 访问单元。

Consecutive access units containing non-reference pictures shall be ordered in ascending order of picture order count.

包含非参考图像的连续访问单元应该按照图像顺序数 POC（picture order count）的升序进行排列。

When present, an access unit following an access unit that contains an end of sequence NAL unit shall be an IDR access unit.

在包含序列最后一个 NAL 单元的访问单元之后的访问单元应该是一个 IDR 访问单元。

When an SEI NAL unit contains data that pertain to more than one access unit (for example, when the SEI NAL unit has a coded video sequence as its scope), it shall be contained in the first access unit to which it applies.

当一个 SEI NAL 单元中包含属于多个访问单元的数据时（例如：当 SEI NAL 单元有一个编码视频序列作为它的范围），它将被包含在被应用的第一个访问单元中。

When an end of stream NAL unit is present in an access unit, this access unit shall be the last NAL unit in the bitstream.

在访问单元中出现 NAL 单元流的结束时，那么这个访问单元可能是码流中的最后一个访问单元。

7.4.1.2.3 Order of NAL units and coded pictures and association to access units

7.4.1.2.3 NAL 单元的顺序和编码图像和访问单元的关系

An access unit consists of one primary coded picture, zero or more corresponding redundant coded pictures, and zero or more non-VCL NAL units. The association of VCL NAL units to primary or redundant coded pictures is described in subclause 7.4.1.2.5.

一个访问单元由一个基本编码图像、零个或多个对应的冗余编码图像，和零个或多个 non-VCL NAL 单元组成。VCL NAL 单元和基本编码图像或者冗余编码图像之间的关系在 7.4.1.2.5 小节中进行描述。

The first of any of the following NAL units after the last VCL NAL unit of a primary coded picture specifies the start of a new access unit.

基本编码图像的最后一个 VCL NAL 单元之后的第一个 NAL 单元表示了一个新访问单元的开始。

- access unit delimiter NAL unit (when present)
- sequence parameter set NAL unit (when present)
- picture parameter set NAL unit (when present)
- SEI NAL unit (when present)
- NAL units with nal_unit_type in the range of 13 to 18, inclusive
- first VCL NAL unit of a primary coded picture (always present)
- 访问单元分割符 NAL 单元（可能出现）
- 序列参数集 NAL 单元（可能出现）
- 图像参数集 NAL 单元（可能出现）
- SEI NAL 单元（可能出现）
- nal_unit_type 等于 13 到18（包括 13 和 18）的 NAL 单元
- 基本编码图像的第一个 VCL NAL 单元（总是出现）

The constraints for the detection of the first VCL NAL unit of a primary coded picture are specified in subclause 7.4.1.2.4.

对于基本编码图像的第一个 VCL NAL 单元的检测的限制在 7.4.1.2.4 小节中说明。

The following constraints shall be obeyed by the order of the coded pictures and non-VCL NAL units within an access unit.

在一个访问单元中，编码图像和 non-VCL NAL 单元的顺序必须遵循下面的限制。

- When an access unit delimiter NAL unit is present, it shall be the first NAL unit. There shall be at most one access unit delimiter NAL unit in any access unit.
- When any SEI NAL units are present, they shall precede the primary coded picture.
- When an SEI NAL unit containing a buffering period SEI message is present, the buffering period SEI message shall be the first SEI message payload of the first SEI NAL unit in the access unit
- The primary coded picture shall precede the corresponding redundant coded pictures.

- 当访问单元的分隔符 NAL 单元出现时，它将是第一个 NAL 单元，在任何一个访问单元中最多有一个访问单元分隔符 NAL 单元。
- 当任何 SEI NAL 出现时，它们都在基本编码图像之前。
- 当含有缓冲周期 SEI 消息的 SEI NAL 单元出现时，缓冲周期 SEI 消息将是访问单元中的第一个 SEI NAL 单元的第一个 SEI 消息负载（负载？）。
- 基本编码图像应该在对应的冗余编码图像之前。
- When redundant coded pictures are present, they shall be ordered in ascending order of the value of redundant_pic_cnt.
- When an end of sequence NAL unit is present, it shall follow the primary coded picture and all redundant coded pictures (if any).
- When an end of stream NAL unit is present, it shall be the last NAL unit.
- sequence parameter set NAL units or picture parameter set NAL units may be present in the access unit.
- NAL units having nal_unit_type equal to 0, 12, or in the range of 19 to 31, inclusive, shall not precede the first VCL NAL unit of the primary coded picture.
- 当出现冗余编码图像时，它们应该按照 redundant_pic_cnt 值的升序进行排列。
- 当出现 NAL 单元序列的结束符时，它将在所有的基本图像和冗余图像之后。
- 当出现 NAL 单元码流的结束符时，它将是最后一个 NAL 单元。
- 序列参数集 NAL 单元或者图像参数集 NAL 单元可能出现在访问单元中。
- nal_unit_type 等于 0、12、或者在19到31范围内（包括19和31 ）的 NAL 单元不应该在基本编码图像的第一个 VCL NAL 单元之前。

NOTE – When a NAL unit having nal_unit_type equal to 7 or 8 is present in an access unit, it may not be referred to in the coded pictures of the access unit in which it is present, and may be referred to in coded pictures of subsequent access units.
注 — 当 nal_unit_type 等于 7 或者 8 的 NAL 单元出现在访问单元中时，它也许不被该访问单元中的编码图像参考，可能被接下来的访问单元中的编码图像参考。

The structure of access units not containing any NAL units with nal_unit_type equal to 0, 7, 8, or in the range of 12 to 31, inclusive, is shown in Figure 7-1.

不含有 nal_unit_type 等于 0、7、8、或者在12到31范围内（包括12 和 31）的任何 NAL 单元的访问单元的结构如图 Figure 7-1 所示。

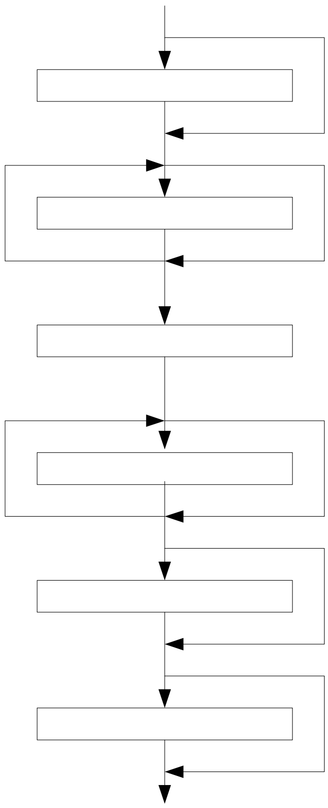


Figure 7-1 – The structure of an access unit not containing any NAL units with nal_unit_type equal to 0, 7, 8, or in the range of 12 to 31, inclusive

7.4.1.2.4 Detection of the first VCL NAL unit of a primary coded picture

7.4.1.2.4 基本编码图像的第一个 VCL NAL 单元的检测

Any coded slice NAL unit or coded slice data partition A NAL unit of the primary coded picture of the current access unit shall be different from any coded slice NAL unit or coded slice data partition A NAL unit of the primary coded picture of the previous access unit in one or more of the following ways.

当前访问单元中基本编码图像的任何一个编码 slice 单元或者编码 slice 数据块 A NAL 单元和以前访问单元中基本编码图像的任何一个编码 slice 单元或者编码 slice 数据块 A NAL 单元在下面一个或几个方面都是不相同的。

- frame_num differs in value.
- field_pic_flag differs in value.
- bottom_field_flag is present in both and differs in value.
- nal_ref_idc differs in value with one of the nal_ref_idc values being equal to 0.
- frame_num is the same for both and pic_order_cnt_type is equal to 0 for both and either pic_order_cnt_lsb differs in value, or delta_pic_order_cnt_bottom differs in value.
- frame_num is the same for both and pic_order_cnt_type is equal to 1 for both and either delta_pic_order_cnt[0] differs in value, or delta_pic_order_cnt[1] differs in value.
- nal_unit_type is equal to 5 for both and idr_pic_id differs in value.
- 不同的 frame_num 值
- 不同的 field_pic_flag 值
- 都有 bottom_field_flag 值，但是不同的 bottom_field_flag 值

- 不同的 nal_ref_idc 值，其中一个 nal_ref_idc 等于 0
- 两者有相同的 frame_num，两者的 pic_order_cnt_type 都等于 0，但是有不同的 pic_order_cnt_lsb 值或者不同的 delta_pic_order_cnt_bottom 值
- 两者有相同的 frame_num，两者的 pic_order_cnt_type 都等于 1，但是有不同的 delta_pic_order_cnt[0] 值或者不同的 delta_pic_order_cnt[1] 值
- 两者有相同的 nal_unit_type 值并且都等于5，但是不同的 idr_pic_id 值

NOTE – Some of the VCL NAL units in redundant coded pictures or some non-VCL NAL units (e.g. an access unit delimiter NAL unit) may also be used for the detection of the boundary between access units, and may therefore aid in the detection of the start of a new primary coded picture.

注 – 一些冗余图像中的 VCL NAL 单元或者一些 non-VCL NAL单元（例如：访问单元分割符 NAL 单元）也可能被用来检测访问单元的边缘和用来检测一个新的基本编码图像的开始

7.4.1.2.5 Order of VCL NAL units and association to coded pictures

7.4.1.2.5 VCL NAL 单元的顺序和与编码图像的关系

Each VCL NAL unit is part of a coded picture.

每个 VCL NAL 单元是编码图像的一部分。

The following constraints shall be obeyed by the order of the VCL NAL units within a coded IDR picture.

在编码 IDR 图像中，VCL NAL 单元的顺序必须遵循下面的约束：

- If arbitrary slice order is allowed as specified in Annex A, coded slice of an IDR picture NAL units may have any order relative to each other.
- Otherwise (arbitrary slice order is not allowed), the order of coded slice of an IDR picture NAL units shall be in the order of increasing macroblock address for the first macroblock of each coded slice of an IDR picture NAL unit.
- 如果在附录 A 中允许使用任意 slice 顺序（arbitrary slice order），IDR 图像的编码 slice NAL 单元可能有任意的顺序。
- 否则（不允许使用任意 slice 顺序），IDR 图像的编码 slice NAL 单元的顺序应该按照每个 slice 中的第一个宏块地址增加的顺序进行排列。

The following constraints shall be obeyed by the order of the VCL NAL units within a coded non-IDR picture.

在编码 non-IDR 图像中，VCL NAL 单元的顺序必须遵循下面的约束：

- If arbitrary slice order is allowed as specified in Annex A, coded slice of a non-IDR picture NAL units or coded slice data partition A NAL units may have any order relative to each other. A coded slice data partition A NAL unit with a particular value of slice_id shall precede any present coded slice data partition B NAL unit with the same value of slice_id. A coded slice data partition A NAL unit with a particular value of slice_id shall precede any present coded slice data partition C NAL unit with the same value of slice_id. If a coded slice data partition B NAL unit with a particular value of slice_id is present, it shall precede any present coded slice data partition C NAL unit with the same value of slice_id.
- 如果使用附录 A 中的任意 slice 顺序，那么 non-IDR 图像的编码 slice NAL 单元或者编码 slice 数据块 A NAL 单元可以使用任意的顺序。带有具体 slice_id 值的编码 slice 数据块 A NAL 单元将出现在其他具有相同 slice_id 值的编码 slice 数据块 B 之前；带有具体 slice_id 值的编码 slice 数据块 A NAL 单元也将出现在其他具有相同 slice_id 值的编码 slice 数据块 C 之前。
- Otherwise (arbitrary slice order is not allowed), the order of coded slice of a non-IDR picture NAL units or coded slice data partition A NAL units shall be in the order of increasing macroblock address for the first macroblock of each coded slice of a non-IDR picture NAL unit or coded slice data partition A NAL unit. A coded slice data partition A NAL unit with a particular value of slice_id shall immediately precede any present coded slice data partition B NAL unit with the same value of slice_id. A coded slice data partition A NAL unit with a particular value of slice_id shall immediately precede any present coded slice data partition C NAL unit with the same value of slice_id, if a coded slice data partition B NAL unit with the same value of slice_id is not present. If a coded slice data partition B NAL unit with a particular value of slice_id is present, it shall immediately precede any present coded slice data partition C NAL unit with the same value of slice_id.

- 否则（不允许使用任意 slice 顺序），non-IDR 图像的编码 slice NAL 单元或者编码 slice 数据块 A NAL 单元的顺序应该按照 non-IDR 图像的编码 slice NAL 单元或者编码 slice 数据块 A NAL 单元中的第一个宏块地址增加的顺序进行排列。带有具体 slice_id 值的编码 slice 数据块 A NAL 单元将立刻出现在其他具有相同 slice_id 值的编码 slice 数据块 B 之前；如果具有相同 slice_id 值的编码 slice 数据块 B不存在，那么带有具体 slice_id 值的编码 slice 数据块 A NAL 单元也将立刻出现在其他具有相同 slice_id 值的编码 slice 数据块 C 之前，如果具有相同 slice_id 值的编码 slice 数据块 B存在，那么它将立刻出现在其他具有相同 slice_id 值的编码 slice 数据块 C 之前。

NAL units having nal_unit_type equal to 0, 12, or in the range of 19 to 31, inclusive, may be present in the access unit but shall not precede the first VCL NAL unit of the primary coded picture within the access unit.

nal_unit_type 等于 0、12、或者在19到31范围内（包括19 和 31）的NAL 单元可能出现在访问单元中，但是不可能出现在访问单元中基本编码图像的第一个 VCL NAL 单元之前。

7.4.2 Raw byte sequence payloads and RBSP trailing bits semantics

7.4.2 原始字节序列负载和 RBSP 后缀比特语法

7.4.2.1 Sequence parameter set RBSP semantics

7.4.2.1 序列参数集 RBSP 语法

profile_idc and **level_idc** indicate the profile and level to which the bitstream conforms, as specified in Annex A.

profile_idc 和 **level_idc** 表示了码流遵循的框架和级，具体见附录 A。

constraint_set0_flag equal to 1 indicates that the bitstream obeys all constraints specified in subclause A.2.1. constraint_set0_flag equal to 0 indicates that the bitstream may or may not obey all constraints specified in subclause A.2.1.

constraint_set0_flag 等于 1 表示码流遵循 A.2.1 小节中的所有限制条件，constraint_set0_flag 等于 0 表示也许遵循或不遵循 A.2.1 小节中的所有限制条件。

constraint_set1_flag equal to 1 indicates that the bitstream obeys all constraints specified in subclause A.2.2. constraint_set1_flag equal to 0 indicates that the bitstream may or may not obey all constraints specified in subclause A.2.2.

constraint_set1_flag 等于 1 表示码流遵循 A.2.2 小节中的所有限制条件，constraint_set0_flag 等于 0 表示也许遵循或不遵循 A.2.2 小节中的所有限制条件。

constraint_set2_flag equal to 1 indicates that the bitstream obeys all constraints specified in subclause A.2.3. constraint_set2_flag equal to 0 indicates that the bitstream may or may not obey all constraints specified in subclause A.2.3.

constraint_set2_flag 等于 1 表示码流遵循 A.2.3 小节中的所有限制条件，constraint_set0_flag 等于 0 表示也许遵循或不遵循 A.2.3 小节中的所有限制条件。

NOTE – When more than one of constraint_set0_flag, constraint_set1_flag, or constraint_set2_flag are equal to 1, the bitstream obeys the constraints of all of the indicated subclauses of subclause A.2.

注 — 当 constraint_set0_flag、constraint_set1_flag 或 constraint_set2_flag 中的多个等于 1 时，那么比特流将遵循 A.2 小节中表示的所有限制。

reserved_zero_5bits shall be equal to 0 in bitstreams conforming to this Recommendation | International Standard. Other values of reserved_zero_5bits may be specified in the future by ITU-T | ISO/IEC. Decoders shall ignore the value of reserved_zero_5bits.

reserved_zero_5bits 在遵循协议的比特流中将等于 0。reserved_zero_5bits 的其他值也许将来由 ITU-T | ISO/IEC 来说明，解码器应该忽略 reserved_zero_5bits 的值。

seq_parameter_set_id identifies the sequence parameter set that is referred to by the picture parameter set. The value of seq_parameter_set_id shall be in the range of 0 to 31, inclusive.

seq_parameter_set_id 表示了图像参数集参考的序列参数集。seq_parameter_set_id 的值应该在 0 到 31 之间，包括 0 和 31。

NOTE – When feasible, encoders should use distinct values of seq_parameter_set_id when the values of other sequence parameter set syntax elements differ rather than changing the values of the syntax elements associated with a specific value of seq_parameter_set_id.

注 — 如果可行的话，当其他的序列参数集语法元素不相同时，编码器应该使用不同的 seq_parameter_set_id 值而不是改变和某一 seq_parameter_set_id 相关的语法元素。

log2_max_frame_num_minus4 specifies the value of the variable MaxFrameNum that is used in frame_num related derivations as follows:

log2_max_frame_num_minus4 说明了变量 MaxFrameNum 的值，这个值在和 frame_num 相关的计算中使用。如下所示

$$\text{MaxFrameNum} = 2^{(\text{ log2\_max\_frame\_num\_minus4} + 4)} \qquad (7\text{-}1)$$

The value of log2_max_frame_num_minus4 shall be in the range of 0 to 12, inclusive.

log2_max_frame_num_minus4 值的范围是 0 到 12，包括 0 和 12。

pic_order_cnt_type specifies the method to decode picture order count (as specified in subclause 8.2.1). The value of pic_order_cnt_type shall be in the range of 0 to 2, inclusive.

pic_order_cnt_type 说明了解码图像顺序数的方法（如 8.2.1 小节所示），pic_order_cnt_type 值的范围应该在 0 到 2 之间，包括 0 和 2。

pic_order_cnt_type shall not be equal to 2 in a coded video sequence that contains any of the following

在下面的编码视频序列中 pic_order_cnt_type 的值不等于 2。

- an access unit containing a non-reference frame followed immediately by an access unit containing a non-reference picture
- two access units each containing a field with the two fields together forming a non-reference field pair followed immediately by an access unit containing a non-reference picture
- an access unit containing a non-reference field followed immediately by an access unit containing another non-reference picture that does not form a complementary field pair with the first of the two access units
- 一个含有非参考帧的访问单元后面接着又是一个含有非参考帧的访问单元
- 两个访问单元（每个访问单元含有两场中的一场，这两场可以合并成一非参考场对）后面接着又是一个含有非参考图像的访问单元
- 一个含有非参考场的访问单元后面接着又是一个含有另外一个非参考图像的访问单元，但是这个非参考图像和前面的非参考场不能够形成一个互补的场对

log2_max_pic_order_cnt_lsb_minus4 specifies the value of the variable MaxPicOrderCntLsb that is used in the decoding process for picture order count as specified in subclause 8.2.1 as follows:

log2_max_pic_order_cnt_lsb_minus4 表示了变量 MaxPicOrderCntLsb 的值，这个值在解码图像顺序数的过程中使用，如 8.2.1 小节所示。

$$\text{MaxPicOrderCntLsb} = 2^{(\text{ log2\_max\_pic\_order\_cnt\_lsb\_minus4} + 4)} \qquad (7\text{-}2)$$

The value of log2_max_pic_order_cnt_lsb_minus4 shall be in the range of 0 to 12, inclusive.

log2_max_pic_order_cnt_lsb_minus4 值的范围是 0 到 12 ，包括 0 和 12 。

delta_pic_order_always_zero_flag equal to 1 specifies that delta_pic_order_cnt[0] and delta_pic_order_cnt[1] are not present in the slice headers of the sequence and shall be inferred to be equal to 0. delta_pic_order_always_zero_flag equal to 0 specifies that delta_pic_order_cnt[0] is present in the slice headers of the sequence and delta_pic_order_cnt[1] may be present in the slice headers of the sequence.

delta_pic_order_always_zero_flag 等于 1 表示在序列的 slice头信息中不会出现 delta_pic_order_cnt[0] 和 delta_pic_order_cnt[1]，而它们的值将等于 0。delta_pic_order_always_zero_flag 等于 0 表示在序列的 slice头信息中会出现 delta_pic_order_cnt[0] 和 delta_pic_order_cnt[1] 。

offset_for_non_ref_pic is used to calculate the picture order count of a non-reference picture as specified in 8.2.1. The value of offset_for_non_ref_pic shall be in the range of -2^{31} to $2^{31} - 1$, inclusive.

offset_for_non_ref_pic 被用来计算非参考图像的图像顺序数，如 8.2.1 小节所示。offset_for_non_ref_pic 值的范围是 -2^{31} 到 $2^{31} - 1$，包括 -2^{31} 和 $2^{31} - 1$。

offset_for_top_to_bottom_field is used to calculate the picture order count of the bottom field in a frame as specified in 8.2.1. The value of offset_for_top_to_bottom_field shall be in the range of -2^{31} to $2^{31} - 1$, inclusive.

offset_for_top_to_bottom_field 被用来计算在一帧中底场的图像顺序数，如 8.2.1 小节所示，offset_for_top_to_bottom_field 值的范围是 -2^{31} 到 $2^{31}-1$，包括 -2^{31} 和 $2^{31}-1$。

num_ref_frames_in_pic_order_cnt_cycle is used in the decoding process for picture order count as specified in subclause 8.2.1. The value of num_ref_frames_in_pic_order_cnt_cycle shall be in the range of 0 to 255.

num_ref_frames_in_pic_order_cnt_cycle 在解码图像顺序数的过程中使用，如 8.2.1 小节所示，num_ref_frames_in_pic_order_cnt_cycle 值的范围是 0 到 255，包括 0 和 255。

offset_for_ref_frame[i **]** is an element of a list of num_ref_frames_in_pic_order_cnt_cycle values used in the decoding process for picture order count as specified in subclause 8.2.1. The value of offset_for_ref_frame[i] shall be in the range of -2^{31} to $2^{31}-1$, inclusive.

offset_for_ref_frame[i **]** 是 num_ref_frames_in_pic_order_cnt_cycle 个值的列表中的一个元素，在解码图像顺序数的过程中使用，如 8.2.1 小节所示，offset_for_ref_frame[i] 值的范围是 -2^{31} 到 $2^{31}-1$，包括 -2^{31} 和 $2^{31}-1$。

num_ref_frames specifies the maximum total number of short-term and long-term reference frames, complementary reference field pairs, and non-paired reference fields used by the decoding process for inter prediction of any picture in the sequence. num_ref_frames also determines the size of the sliding window operation as specified in subclause 8.2.5.3. The value of num_ref_frames shall be in the range of 0 to 16, inclusive.

num_ref_frames 表示了最大数量的短期和长期参考图像、互补参考场对和非成对的参考场，这些参考图像在序列中图像的帧间预测中都可能使用。num_ref_frames 也决定了 8.2.5.3 小节中滑动窗口操作的大小。num_ref_frames 值的范围是 0 到 16，包括 0 和 16。

gaps_in_frame_num_value_allowed_flag specifies the allowed values of frame_num as specified in subclause 7.4.3 and the decoding process in case of an inferred gap between values of frame_num as specified in subclause 8.2.5.2.

gaps_in_frame_num_value_allowed_flag 表示了 7.4.3 小节中说明的 frame_num 被允许的值，和 8.2.5.2 小节的在 frame_num 之间推测间隔的解码过程中 frame_num 被允许的值。

pic_width_in_mbs_minus1 plus 1 specifies the width of each decoded picture in units of macroblocks.

pic_width_in_mbs_minus1 加 1 表示了每一个解码图像以宏块为单位的宽度。

The variable for the picture width in units of macroblocks is derived as follows

以宏块为单位的图像宽度的变量计算如下：

$$\text{PicWidthInMbs} = \text{pic\_width\_in\_mbs\_minus1} + 1 \quad (7\text{-}3)$$

The variable for picture width for the luma component is derived as follows

亮度分量图像宽度的变量计算如下：

$$\text{PicWidthInSamples}_L = \text{PicWidthInMbs} * 16 \quad (7\text{-}4)$$

The variable for picture width for the chroma components is derived as follows

色差分量图像宽度的变量计算如下：

$$\text{PicWidthInSamples}_C = \text{PicWidthInMbs} * 8 \quad (7\text{-}5)$$

pic_height_in_map_units_minus1 plus 1 specifies the height in slice group map units of a decoded frame or field.

pic_height_in_map_units_minus1 加 1 表示了解码帧或解码场以 slice 组映射单元为单位的高度。

The variables PicHeightInMapUnits and PicSizeInMapUnits are derived as follows

变量 PicHeightInMapUnits 和 PicSizeInMapUnits 的计算如下：

$$\text{PicHeightInMapUnits} = \text{pic\_height\_in\_map\_units\_minus1} + 1 \quad (7\text{-}6)$$

$$\text{PicSizeInMapUnits} = \text{PicWidthInMbs} * \text{PicHeightInMapUnits} \quad (7\text{-}7)$$

frame_mbs_only_flag equal to 0 specifies that coded pictures of the coded video sequence may either be coded fields or coded frames. frame_mbs_only_flag equal to 1 specifies that every coded picture of the coded video sequence is a coded frame containing only frame macroblocks.

frame_mbs_only_flag 等于 0 表示了编码视频序列中的编码图像可以是编码场也可以是编码帧。frame_mbs_only_flag 等于 1 表示编码视频序列中的编码图像是编码帧，而且这些帧中仅仅含有帧宏块。

The allowed range of values for pic_width_in_mbs_minus1, pic_height_in_map_units_minus1, and frame_mbs_only_flag is specified by constraints in Annex A.

pic_width_in_mbs_minus1、pic_height_in_map_units_minus1 和 frame_mbs_only_flag 值的允许范围由附录 A 中的限制说明。

Depending on frame_mbs_only_flag the following semantics are assigned to pic_height_in_map_units_minus1.

根据 frame_mbs_only_flag 的值，下面的语义被分配给 pic_height_in_map_units_minus1。

- If frame_mbs_only_flag is equal to 0, pic_height_in_map_units_minus1 is the height of a field in units of macroblocks.
- Otherwise (frame_mbs_only_flag is equal to 1), pic_height_in_map_units_minus1 is the height of a frame in units of macroblocks.

- 如果 frame_mbs_only_flag 等于 0，pic_height_in_map_units_minus1 是以宏块为单位的场的高度。
- 否则（frame_mbs_only_flag 等于 1），pic_height_in_map_units_minus1 是以宏块为单位的帧的高度。

The variable FrameHeightInMbs is derived as follows

变量 FrameHeightInMbs 的计算如下：

$$\text{FrameHeightInMbs} = (2 - \text{frame\_mbs\_only\_flag}) * \text{PicHeightInMapUnits} \quad (7\text{-}8)$$

mb_adaptive_frame_field_flag equal to 0 specifies no switching between frame and field macroblocks within a picture. mb_adaptive_frame_field_flag equal to 1 specifies the possible use of switching between frame and field macroblocks within frames. When mb_adaptive_frame_field_flag is not present, it shall be inferred to be equal to 0.

mb_adaptive_frame_field_flag 等于 0 表示在图像中没有帧宏块和场宏块之间的切换。mb_adaptive_frame_field_flag 等于 1 表示在图像中可能由帧宏块和场宏块之间的切换。当码流中没有出现 mb_adaptive_frame_field_flag 时，它的值将被设置成 0。

direct_8x8_inference_flag specifies the method used in the derivation process for luma motion vectors for B_Skip, B_Direct_16x16 and B_Direct_8x8 as specified in subclause 8.4.1.2. When frame_mbs_only_flag is equal to 0, direct_8x8_inference_flag shall be equal to 1.

direct_8x8_inference_flag 表示了对于 B_Skip、B_Direct_16x16 和 B_Direct_8x8 类型计算亮度运动矢量过程中使用的方法，如 8.4.1.2 小节所示。当 frame_mbs_only_flag 等于 0 时，direct_8x8_inference_flag 的值将等于 1。

frame_cropping_flag equal to 1 specifies that the frame cropping offset parameters follow next in the sequence parameter set. frame_cropping_flag equal to 0 specifies that the frame cropping offset parameters are not present.

frame_cropping_flag 等于 1 表示了在序列参数集中将出现帧裁剪偏移参数。frame_cropping_flag 等于 0表示了在序列参数集中不出现帧裁剪偏移参数。

frame_crop_left_offset, frame_crop_right_offset, frame_crop_top_offset, frame_crop_bottom_offset specify the samples of a frame within a rectangle as follows.

frame_crop_left_offset, frame_crop_right_offset, frame_crop_top_offset, frame_crop_bottom_offset 表示了矩形内帧的象素。

– If frame_mbs_only_flag is equal to 1, the cropping rectangle contains luma samples with horizontal coordinates from 2 * frame_crop_left_offset to $\text{PicWidthInSamples}_L$ - (2 * frame_crop_right_offset + 1) and vertical coordinates from 2 * frame_crop_top_offset to (FrameHeightInMbs * 16) - (2 * frame_crop_bottom_offset + 1), inclusive. In this case, the value of frame_crop_left_offset shall be in the range of 0 to 8 * PicWidthInMbs - (frame_crop_right_offset + 1), inclusive; and the value of frame_crop_top_offset shall be in the range of 0 to 8 * FrameHeightInMbs - (frame_crop_bottom_offset + 1), inclusive.

– 如果 frame_mbs_only_flag 等于 1，裁剪矩形中包含了水平坐标从 2 * frame_crop_left_offset 到 $\text{PicWidthInSamples}_L$ - (2 * frame_crop_right_offset + 1) 和垂直坐标从 2 * frame_crop_top_offset 到 (FrameHeightInMbs * 16) - (2 * frame_crop_bottom_offset + 1) 的亮度象素，包括边缘的象素。在这种情况下，frame_crop_left_offset 值的范围是从 0 到 8 * PicWidthInMbs - (frame_crop_right_offset + 1)，并且包括这两个数，frame_crop_top_offset 值的范围是从 0 到 8 * FrameHeightInMbs - (frame_crop_bottom_offset + 1)，并且也包括这两个数。（这里的范围好像有错误）

– Otherwise (frame_mbs_only_flag is equal to 0), the cropping rectangle contains luma samples with horizontal coordinates from 2 * frame_crop_left_offset to $PicWidthInSamples_L$ - (2 * frame_crop_right_offset + 1) and vertical coordinates from 4 * frame_crop_top_offset to (FrameHeightInMbs * 16) - (4 * frame_crop_bottom_offset + 1), inclusive. In this case the value of frame_crop_left_offset shall be in the range of 0 to 8 * PicWidthInMbs - (frame_crop_right_offset + 1), inclusive; and the value of frame_crop_top_offset shall be in the range of 0 to 4 * FrameHeightInMbs - (frame_crop_bottom_offset + 1), inclusive.

– 否则（ frame_mbs_only_flag 等于 0），裁剪矩形中包含水平坐标位置从 2 * frame_crop_left_offset 到 $PicWidthInSamples_L$ - (2 * frame_crop_right_offset + 1) 和 垂直坐标位置从 4 * frame_crop_top_offset 到 (FrameHeightInMbs * 16) - (4 * frame_crop_bottom_offset + 1)的亮度象素，包括边缘象素。在这种情况下，frame_crop_left_offset 值的范围应该是 0 到 8 * PicWidthInMbs - (frame_crop_right_offset + 1)，包括这两个数，frame_crop_top_offset 值的范围应该是 0 到 4 * FrameHeightInMbs - (frame_crop_bottom_offset + 1), 包括这两个数。（这里的范围好像有错误）

When frame_cropping_flag is equal to 0, the following values shall be inferred: frame_crop_left_offset = 0, frame_crop_right_offset = 0, frame_crop_top_offset = 0, and frame_crop_bottom_offset = 0.

当 frame_cropping_flag 等于 0 时，将使用下面的值：frame_crop_left_offset = 0, frame_crop_right_offset = 0, frame_crop_top_offset = 0, and frame_crop_bottom_offset = 0。

The specified samples of the two chroma arrays are the samples having frame coordinates (x / 2, y / 2), where (x, y) are the frame coordinates of the specified luma samples.

两个色差数组的象素是帧坐标为(x / 2, y / 2)的象素，这里 (x, y) 是帧亮度象素的帧坐标。

For decoded fields, the specified samples of the decoded field are the samples that fall within the rectangle specified in frame coordinates.

对于解码场，解码场的象素是位于帧坐标矩形内的象素。

vui_parameters_present_flag equal to 1 specifies that the vui_parameters( ) syntax structure specified in Annex E is present next in the bitstream. vui_parameters_present_flag equal to 0 specifies that the vui_parameters( ) syntax structure specified in Annex E is not present next in the bitstream.

vui_parameters_present_flag 等于 1 表示在下面的码流中将出现附录 E 说明的 vui_parameters( ) 语法结构。vui_parameters_present_flag 等于 0 表示在下面的码流中不出现附录 E 说明的 vui_parameters( ) 语法结构。

7.4.2.2 Picture parameter set RBSP semantics

7.4.2.2 图像参数集 RBSP 的语义

pic_parameter_set_id identifies the picture parameter set that is referred to in the slice header. The value of pic_parameter_set_id shall be in the range of 0 to 255, inclusive.

pic_parameter_set_id 表示了在 slice 头中参考的图像参数集，pic_parameter_set_id 值的范围是 0 到 255，包括 0 和 255。

seq_parameter_set_id refers to the active sequence parameter set. The value of seq_parameter_set_id shall be in the range of 0 to 31, inclusive.

seq_parameter_set_id 表示了参考的有效序列参数集，seq_parameter_set_id 值的范围是 0 到 31，包括 0 和 31。

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows.

entropy_coding_mode_flag 选择了对于语法元素的熵解码模式，对于不同的熵解码模式，在语法表中将出现两种不同的描述符。

- If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see subclause 9.1 or CAVLC, see subclause 9.2).

- Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see subclause 9.3).

- 如果 entropy_coding_mode_flag 等于 0，就使用语法表中左边的描述符表示的方法（Exp-Golomb 编码, 见 9.1 小节 或者 CAVLC, 见 9.2 小节）。

- 否则（entropy_coding_mode_flag 等于 1），就使用语法表中右边的描述符表示的方法（CABAC，见 9.3 ）

pic_order_present_flag equal to 1 specifies that the picture order count related syntax elements are present in the slice headers as specified in subclause 7.3.3. pic_order_present_flag equal to 0 specifies that the picture order count related syntax elements are not present in the slice headers.

pic_order_present_flag 等于 1 表示在 slice 头信息中将出现与图像顺序数相关的语法元素，如 7.3.3 小节所示。pic_order_present_flag 等于 0 表示在 slice 头信息中不出现与图像顺序数相关的语法元素。

num_slice_groups_minus1 plus 1 specifies the number of slice groups for a picture. When num_slice_groups_minus1 is equal to 0, all slices of the picture belong to the same slice group. The allowed range of num_slice_groups_minus1 is specified in Annex A.

num_slice_groups_minus1 加 1 表示了图像中 slice 组的数目。当 num_slice_groups_minus1 等于 0，图像中所有的 slice 属于相同的 slice 组。num_slice_groups_minus1 值的范围在附录 A 中规定。

slice_group_map_type specifies how the mapping of slice group map units to slice groups is coded. The value of slice_group_map_type shall be in the range of 0 to 6, inclusive.

slice_group_map_type 表示了 slice 组映射单元到 slice 组的映射方式。slice_group_map_type 值的范围是 0 到 6，包括 0 和 6。

slice_group_map_type equal to 0 specifies interleaved slice groups.

slice_group_map_type equal to 1 specifies a dispersed slice group mapping.

slice_group_map_type equal to 2 specifies one or more “foreground” slice groups and a “leftover” slice group.

slice_group_map_type 等于 0 表示了交叉 slice 组。

slice_group_map_type 等于 1 表示了分散的 slice 组映射。

slice_group_map_type 等于 2 表示了一个或多个“前景”slice 组和一个“剩余”slice 组。

slice_group_map_type values equal to 3, 4, and 5 specify changing slice groups. When num_slice_groups_minus1 is not equal to 1, slice_group_map_type shall not be equal to 3, 4, or 5.

slice_group_map_type equal to 6 specifies an explicit assignment of a slice group to each slice group map unit.

slice_group_map_type 等于 3、4 和 5 表示了可变 slice 组，当 num_slice_groups_minus1 不等于 1 时，slice_group_map_type 也不等于 3、4 或者 5。

slice_group_map_type 等于 6 表示了一个由 slice 组到 slice 组映射单元的显式分配过程。

Slice group map units are specified as follows:

Slice 组映射单元的说明如下：

– If frame_mbs_only_flag is equal to 0 and mb_adaptive_frame_field_flag is equal to 1 and the coded picture is a frame, the slice group map units are macroblock pair units.
– If frame_mbs_only_flag is equal to 1 or a coded picture is a field, the slice group map units are units of macroblocks.
– Otherwise (frame_mbs_only_flag is equal to 0 and mb_adaptive_frame_field_flag is equal to 0 and the coded picture is a frame), the slice group map units are units of two macroblocks that are vertically contiguous as in a frame macroblock pair of an MBAFF frame.
– 如果 frame_mbs_only_flag 等于 0 和 mb_adaptive_frame_field_flag 等于 1 和 编码图像是帧图像，那么 slice 组映射单元是宏块对单元。
– 如果 frame_mbs_only_flag 等于 1 或者 编码图像是场图像，那么 slice 组映射单元是宏块对单元。
– 否则（frame_mbs_only_flag 等于 0 和 mb_adaptive_frame_field_flag 等于 0 和 编码图像是帧图像），slice 组映射单元是垂直相临的两个宏块单元，象宏块帧场自适应（ MBAFF） 帧中的帧宏块对。

run_length_minus1[i **]** is used to specify the number of consecutive slice group map units to be assigned to the i-th slice group in raster scan order of slice group map units. The value of run_length_minus1[i] shall be in the range of 0 to PicSizeInMapUnits - 1, inclusive.

run_length_minus1[i **]** 表示按照 slice 组映射单元的栅格扫描顺序分配给第 i 个slice 组的连续 slice 组映射单元的数目。run_length_minus1[i] 值的范围应该是 0 到 PicSizeInMapUnits – 1，包括这两个数。

top_left[i **]** and **bottom_right[** i **]** specify the top-left and bottom-right corners of a rectangle, respectively. top_left[i] and bottom_right[i] are slice group map unit positions in a raster scan of the picture for the slice group map units. For each rectangle i, all of the following constraints shall be obeyed by the values of the syntax elements top_left[i] and bottom_right[i]

top_left[i] 和 **bottom_right[i]** 分别表示了一个矩形的左上角和右下角坐标。top_left[i] 和 bottom_right[i] 是按照 slice 组映射单元栅格扫描顺序的 slice 组映射单元的位置。对于每一个矩形 i ，语法元素 top_left[i] 和 bottom_right[i] 的值必须遵循下面的限制。

- top_left[i] shall be less than or equal to bottom_right[i] and bottom_right[i] shall be less than PicSizeInMapUnits.
- (top_left[i] % PicWidthInMbs) shall be less than or equal to the value of (bottom_right[i] % PicWidthInMbs).
- top_left[i] 应该小于等于 bottom_right[i]，并且 bottom_right[i] 应该小于 PicSizeInMapUnits。
- (top_left[i] % PicWidthInMbs) 应该小于或者等于 (bottom_right[i] % PicWidthInMbs)。

slice_group_change_direction_flag is used with slice_group_map_type to specify the refined map type when slice_group_map_type is 3, 4, or 5.

slice_group_change_direction_flag 被用来和 slice_group_map_type 一起表示改进的映射类型，当 slice_group_map_type 等于 3、4 或者 5 时。

slice_group_change_rate_minus1 is used to specify the variable SliceGroupChangeRate. SliceGroupChangeRate specifies the multiple in number of slice group map units by which the size of a slice group can change from one picture to the next. The value of slice_group_change_rate_minus1 shall be in the range of 0 to PicSizeInMapUnits – 1, inclusive. The SliceGroupChangeRate variable is specified as follows:

slice_group_change_rate_minus1 说明了变量 SliceGroupChangeRate 的值。SliceGroupChangeRate 表示了 slice 组映射单元的数目，根据这个数目，slice 组的大小从一个图像到下一个图像可以改变。slice_group_change_rate_minus1 值的范围是 0 到 PicSizeInMapUnits – 1，包括这两个数。变量 SliceGroupChangeRate 的说明如下：

$$\text{SliceGroupChangeRate} = \text{slice\_group\_change\_rate\_minus1} + 1 \tag{7-9}$$

pic_size_in_map_units_minus1 is used to specify the number of slice group map units in the picture. pic_size_in_map_units_minus1 shall be equal to PicSizeInMapUnits - 1.

pic_size_in_map_units_minus1 被用来表示图像中 slice 映射单元的数目。pic_size_in_map_units_minus1 应该等于 PicSizeInMapUnits – 1。

slice_group_id[i] identifies a slice group of the i-th slice group map unit in raster scan order. The size of the slice_group_id[i] syntax element is Ceil(Log2(num_slice_groups_minus1 + 1)) bits. The value of slice_group_id[i] shall be in the range of 0 to num_slice_groups_minus1, inclusive.

slice_group_id[i] 表示了按照栅格扫描顺序第 i 个 slice 组映射单元的 slice 组。语法元素 slice_group_id[i] 的大小是 Ceil(Log2(num_slice_groups_minus1 + 1)) 比特，slice_group_id[i] 的值应该在 0 到 num_slice_groups_minus1 之间，包括这两个数。

num_ref_idx_l0_active_minus1 specifies the maximum reference index for reference picture list 0 that shall be used to decode each slice of the picture in which list 0 is used if num_ref_idx_active_override_flag is equal to 0 for the slice. When MbaffFrameFlag is equal to 1, num_ref_idx_l0_active_minus1 is the maximum index value for the decoding of frame macroblocks and 2 * num_ref_idx_l0_active_minus1 + 1 is the maximum index value for the decoding of field macroblocks. The value of num_ref_idx_l0_active_minus1 shall be in the range of 0 to 31, inclusive.

num_ref_idx_l0_active_minus1 表示了参考图像列表 0 中的最大参考索引，对于 num_ref_idx_active_override_flag 等于 0 的 slice 来说，图像列表 0 将被用来解码使用列表 0 的图像中的每一个 slice。当 MbaffFrameFlag 等于 1 时，num_ref_idx_l0_active_minus1 是解码帧宏块的最大索引值，而 2 * num_ref_idx_l0_active_minus1 + 1 是解码场宏块的最大索引值。num_ref_idx_l0_active_minus1 值的范围是 0 到 31，包括 0 和 31。

num_ref_idx_l1_active_minus1 has the same semantics as num_ref_idx_l0_active_minus1 with l0 and list 0 replaced by l1 and list 1, respectively.

num_ref_idx_l1_active_minus1 有和 num_ref_idx_l0_active_minus1 有相同的语义，l0 和列表 0 由 l1 和列表 1 来分别替代。

weighted_pred_flag equal to 0 specifies that weighted prediction shall not be applied to P and SP slices. weighted_pred_flag equal to 1 specifies that weighted prediction shall be applied to P and SP slices.

weighted_pred_flag 等于 0 表示了对 P 和 SP slice 不使用加权预测。weighted_pred_flag 等于 1 表示了对 P 和 SP slice 使用加权预测。

weighted_bipred_idc equal to 0 specifies that the default weighted prediction shall be applied to B slices. weighted_bipred_idc equal to 1 specifies that explicit weighted prediction shall be applied to B slices. weighted_bipred_idc equal to 2 specifies that implicit weighted prediction shall be applied to B slices. The value of weighted_bipred_idc shall be in the range of 0 to 2, inclusive.

weighted_bipred_idc 等于 0 表示对 B slices 使用缺省的加权预测。weighted_bipred_idc 等于 1 表示对 B slices 使用显式加权预测。weighted_bipred_idc 等于 2 表示对 B slices 使用隐式加权预测。weighted_bipred_idc 值的范围是 0 到 2，包括 0 和 2。

pic_init_qp_minus26 specifies the initial value minus 26 of $SliceQP_Y$ for each slice. The initial value is modified at the slice layer when a non-zero value of slice_qp_delta is decoded, and is modified further when a non-zero value of mb_qp_delta is decoded at the macroblock layer. The value of pic_init_qp_minus26 shall be in the range of -26 to +25, inclusive.

pic_init_qp_minus26 表示了每一个 slice 的 $SliceQP_Y$ 初始值减去 26 的值。当在 slice 层解码到非零的 slice_qp_delta 值时，修改这个初始值，并且如果在宏块层解码到非零的 mb_qp_delta 值时，进一步的修改这个值。pic_init_qp_minus26 值的范围是 -26 到 +25，包括 -26 和 +25 这两个值。

pic_init_qs_minus26 specifies the initial value minus 26 of $SliceQS_Y$ for all macroblocks in SP or SI slices. The initial value is modified at the slice layer when a non-zero value of slice_qs_delta is decoded. The value of pic_init_qs_minus26 shall be in the range of -26 to +25, inclusive.

pic_init_qs_minus26 表示了在 SP 或者 SI slice 中所有宏块的 $SliceQS_Y$ 初始值减去 26 的值。如果在 slice 层解码到非零的 slice_qs_delta 值，将修改这个值。pic_init_qs_minus26 值的范围是 -26 到 +25，包括 -26 和 +25 。

chroma_qp_index_offset specifies the offset that shall be added to QP_Y and QS_Y for addressing the table of QP_C values. The value of chroma_qp_index_offset shall be in the range of -12 to +12, inclusive.

chroma_qp_index_offset 表示了索引 QP_C 值的表时加到 QP_Y 和 QS_Y上的偏移量。chroma_qp_index_offset 值的范围是 -12 到 +12，包括 -12 和 +12 这两个数。

deblocking_filter_control_present_flag equal to 1 specifies that a set of syntax elements controlling the characteristics of the deblocking filter is present in the slice header. deblocking_filter_control_present_flag equal to 0 specifies that the set of syntax elements controlling the characteristics of the deblocking filter is not present in the slice headers and their inferred values are in effect.

deblocking_filter_control_present_flag 等于 1 表示在 slice 头信息中将出现控制环路滤波器特性的语法元素集。deblocking_filter_control_present_flag 等于 0 表示在 slice 头信息中不出现控制环路滤波器特性的语法元素集，它们的推论值有效。

constrained_intra_pred_flag equal to 0 specifies that intra prediction allows usage of neighbouring inter macroblock residual data and decoded samples for the prediction of intra macroblocks, whereas constrained_intra_pred_flag equal to 1 specifies constrained intra prediction, where intra prediction only uses residual data and decoded samples from I or SI macroblock types.

constrained_intra_pred_flag 等于 0 表示了帧内预测过程中可以使用相邻帧间宏块的残差数据和解码象素值来进行帧内宏块的预测，constrained_intra_pred_flag 等于 1 表示使用限制帧内宏块预测，这样帧内预测只能使用 I 和 SI 类型宏块的残差数据和解码象素数据来做预测。

redundant_pic_cnt_present_flag equal to 0 specifies that the redundant_pic_cnt syntax element is not present in slice headers, data partitions B, and data partitions C that refer (either directly or by association with a corresponding data partition A) to the picture parameter set. redundant_pic_cnt_present_flag equal to 1 specifies that the redundant_pic_cnt syntax element is present in all slice headers, data partitions B, and data partitions C that refer (either directly or by association with a corresponding data partition A) to the picture parameter set.

redundant_pic_cnt_present_flag 等于 0 表示在 slice 头信息中、参考图像参数集（或者直接参考或者和对应的数据块 A 相关联）的数据块 B 和 数据块 C 中都没有 redundant_pic_cnt 语法元素。redundant_pic_cnt_present_flag 等于 1 表示在 slice 头信息中、参考图像参数集（或者直接参考或者和对应的数据块 A 相关联）的数据块 B 和数据块 C 中都有 redundant_pic_cnt 语法元素。

7.4.2.3 Supplemental enhancement information RBSP semantics

7.4.2.3 补充增强信息 RBSP 的语义

Supplemental Enhancement Information (SEI) contains information that is not necessary to decode the samples of coded pictures from VCL NAL units.

补充增强信息（SEI）中包含的信息在从 VCL NAL 单元中解码编码图像的象素时不是必须的。

7.4.2.3.1 Supplemental enhancement information message semantics

7.4.2.3.1 补充增强信息消息的语义

An SEI NAL unit contains one or more SEI messages. Each SEI message consists of the variables specifying the type payloadType and size payloadSize of the SEI payload. SEI payloads are specified in Annex D. The derived SEI payload size payloadSize is specified in bytes and shall be equal to the number of bytes in the SEI payload.

每个 SEI NAL 单元中包含一个或者多个 SEI 消息。每个 SEI 消息由说明 SEI 负载的类型和大小的变量 payloadType 和 payloadSize 组成。SEI 负载在附录 D 中说明。计算得到的 SEI 负载的大小是以 字节为单位的，应该等于 SEI 负载的字节数。

ff_byte is a byte equal to 0xFF identifying a need for a longer representation of the syntax structure that it is used within.

ff_byte 是一个等于 0xFF 的字节，表示需要使用一个较长的语法结构。

last_payload_type_byte is the last byte of the payload type of an SEI message.

last_payload_type_byte 是 SEI 消息中负载类型的最后一个字节。

last_payload_size_byte is the last byte of the size of an SEI message.

last_payload_size_byte 是 SEI 消息中负载大写的最后一个字节。

7.4.2.4 Access unit delimiter RBSP semantics

7.4.2.4 访问单元分隔符 RBSP 语义

The access unit delimiter may be used to indicate the type of slices present in a primary coded picture and to simplify the detection of the boundary between access units. There is no normative decoding process associated with the access unit delimiter.

访问单元分隔符可能被用来表示基本编码图像中 slice 的类型和简化访问单元之间边缘的检测。这里没有和访问单元分割符相对应的标准解码过程。

primary_pic_type indicates that the slice_type values for all slices of the primary coded picture are members of the set listed in Table 7-2 for the given value of primary_pic_type.

primary_pic_type 表示了对于基本编码图像中所有 slice 的 slice_type 值，根据给定的 primary_pic_type 值，slice_type 的值如表 Table 7-2 所示。

Table 7-2 – Meaning of primary_pic_type

primary_pic_type	slice_type values that may be present in the primary coded picture
0	I
1	I, P
2	I, P, B
3	SI
4	SI, SP
5	I, SI
6	I, SI, P, SP
7	I, SI, P, SP, B

7.4.2.5 End of sequence RBSP semantics

7.4.2.5 序列结束RBSP 的语义

The end of sequence RBSP specifies that the next subsequent access unit in the bitstream in decoding order (if any) shall be an IDR access unit. The syntax content of the SODB and RBSP for the end of sequence RBSP are empty. No normative decoding process is specified for an end of sequence RBSP.

序列结束 RBSP 表示了按照解码顺序码流中的下一个序列访问单元应该是 IDR 访问单元。对于序列结束 RBSP 来说，SODB 和 RBSP 的语法内容是空的，也没有标准的解码过程。

7.4.2.6 End of stream RBSP semantics

7.4.2.6 流结束 RBSP 的语义

The end of stream RBSP indicates that no additional NAL units shall be present in the bitstream that are subsequent to the end of stream RBSP in decoding order. The syntax content of the SODB and RBSP for the end of stream RBSP are empty. No normative decoding process is specified for an end of stream RBSP.

流结束 RBSP 表示按照解码顺序在码流中流结束 RBSP 的后面没有附加的 NAL 单元。对于流结束 RBSP 来说，SODB 和 RBSP 的语法内容是空的，也没有标准的解码过程。

7.4.2.7 Filler data RBSP semantics

7.4.2.7 填充数据 RBSP 的语义

The filler data RBSP contains bytes whose value shall be equal to 0xFF. No normative decoding process is specified for a filler data RBSP.

填充数据 RBSP 包含有值等于 0xFF 的字节。对于填充数据 RBSP 来说，没有标准的解码过程。

ff_byte is a byte equal to 0xFF.

ff_byte 是一个等于 0xFF 的字节。

7.4.2.8 Slice layer without partitioning RBSP semantics

7.4.2.8 没有分割的 slice 层 RBSP 的语义

The slice layer without partitioning RBSP consists of a slice header and slice data.

没有分割的 slice 层 RBSP 有 slice 头和 slice 数据组成。

7.4.2.9 Slice data partition RBSP semantics

7.4.2.9 slice 数据块 RBSP 的语义

7.4.2.9.1 Slice data partition A RBSP semantics

7.4.2.9.1 Slice 数据块 A RBSP 的语义

When slice data partitioning is in use, the coded data for a single slice is divided into three separate partitions. Partition A contains all syntax elements of category 2.

当使用 slice 数据分割时，每个 slice 的编码数据被分割成三个单独的块，块 A 包含类别 2 的所有语法元素。

Category 2 syntax elements include all syntax elements in the slice header and slice data syntax structures other than the syntax elements in the residual( ) syntax structure.

类别 2 的语法元素包括除了 residual( ) 语法结构中的语法元素之外的slice 头中的所有语法元素和 slice 数据语法结构。

slice_id identifies the slice associated with the data partition. Each slice shall have a unique slice_id value within the coded picture that contains the slice. When arbitrary slice order is not allowed as specified in Annex A, the first slice of a coded picture, in decoding order, shall have slice_id equal to 0 and the value of slice_id shall be incremented by one for each subsequent slice of the coded picture in decoding order.

slice_id 表示了和数据块相关的 slice 。每个 slice 在包含该 slice 的编码图像中有唯一的 slice_id，当不使用附录 A 中的 ASO 时（arbitrary slice order）时，那么按照解码顺序，编码图像中第一个 slice 的 slice_id 应该等于 0，接下来的 slice 的 slice_id 应该依次加 1。

The range of slice_id is specified as follows.

slice_id 的范围说明如下：

– If MbaffFrameFlag is equal to 0, slice_id shall be in the range of 0 to PicSizeInMbs - 1, inclusive.

– Otherwise (MbaffFrameFlag is equal to 1), slice_id shall be in the range of 0 to PicSizeInMbs / 2 - 1, inclusive.

– 如果 MbaffFrameFlag 等于 0，slice_id 值的范围是 0 到 PicSizeInMbs – 1，包括这两个数。

– 否则（MbaffFrameFlag 等于 1），slice_id 值的范围是 0 到 PicSizeInMbs / 2 - 1，包括这两个数。

7.4.2.9.2 Slice data partition B RBSP semantics

7.4.2.9.2 slice 数据块 B RBSP 语义

When slice data partitioning is in use, the coded data for a single slice is divided into one to three separate partitions. Slice data partition B contains all syntax elements of category 3.

当使用 slice 数据分割时，每个 slice 的编码数据被分割成 3 个独立的块。Slice 数据块 B 包含类别 3 的所有语法元素。

Category 3 syntax elements include all syntax elements in the residual( ) syntax structure and in syntax structures used within that syntax structure for collective macroblock types I and SI as specified in Table 7-7.

类别 3 的语法元素包含 residual( ) 语法结构中的所有语法元素和对于 I 和 SI 宏块类型中的 residual( ) 语法结构中使用的语法结构中的所有元素，如表 Table 7-7 所示

slice_id has the same semantics as specified in subclause 7.4.2.9.1.

slice_id 有和 7.4.2.9.1 小节中相同的语义。

redundant_pic_cnt shall be equal to 0 for slices and slice data partitions belonging to the primary coded picture. The redundant_pic_cnt shall be greater than 0 for coded slices and coded slice data partitions in redundant coded pictures. When redundant_pic_cnt is not present, its value shall be inferred to be equal to 0. The value of redundant_pic_cnt shall be in the range of 0 to 127, inclusive.

redundant_pic_cnt 对于属于基本编码图像的 slice 和 slice 数据块来说应该等于 0。在冗余编码图像中，对于编码 slice 和 slice 数据块来说，redundant_pic_cnt 的值应该大于 0。当码流中没有出现 redundant_pic_cnt 时，那么它的值应该被设置为 0。redundant_pic_cnt 的范围是 0 到 127， 包括 0 和 127。

- If the syntax elements of a slice data partition A RBSP indicate the presence of any syntax elements of category 3 in the slice data for a slice, a slice data partition B RBSP shall be present having the same value of slice_id and redundant_pic_cnt as in the slice data partition A RBSP.

- Otherwise (the syntax elements of a slice data partition A RBSP do not indicate the presence of any syntax elements of category 3 in the slice data for a slice), no slice data partition B RBSP shall be present having the same value of slice_id and redundant_pic_cnt as in the slice data partition A RBSP.

- 对于一个 slice 来说，如果 slice 数据块 A RBSP的语法元素表示在 slice 数据中有类别 3 的语法元素，那么就有 slice 数据 B RBSP， 并且和 slice 数据块 A RBSP 有相同的 slice_id 和 redundant_pic_cnt。

- 否则（对于一个 slice 来说，如果 slice 数据块 A RBSP的语法元素表示在 slice 数据中没有类别 3 的语法元素），那么就没有和 slice 数据块 A RBSP 有相同slice_id 和 redundant_pic_cnt 的 slice 数据块 B RBSP。

7.4.2.9.3 Slice data partition C RBSP semantics

7.4.2.9.3 slice 数据块 C RBSP 的语义

When slice data partitioning is in use, the coded data for a single slice is divided into three separate partitions. Slice data partition C contains all syntax elements of category 4.

当使用 slice 数据分割时，每个 slice 的编码数据被分割成 3 个独立的块。Slice 数据块 C 包含类别 4 的所有语法元素。

Category 4 syntax elements include all syntax elements in the residual( ) syntax structure and in syntax structures used within that syntax structure for collective macroblock types P and B as specified in Table 7-7.

类别 4 的语法元素包括 residual( ) 语法结构中的所有语法语法元素和和对于 P 和 B 宏块类型中的 residual( ) 语法结构中使用的语法结构中的所有元素，Table 7-7 所示。

slice_id has the same semantics as specified in subclause 7.4.2.9.1.

slice_id 有和 7.4.2.9.1 小节中相同的语义。

redundant_pic_cnt has the same semantics as specified in subclause 7.4.2.9.2.

redundant_pic_cnt 有和 7.4.2.9.2 小节中相同的语义。

- If the syntax elements of a slice data partition A RBSP indicate the presence of any syntax elements of category 4 in the slice data for a slice, a slice data partition C RBSP shall be present having the same value of slice_id and redundant_pic_cnt as in the slice data partition A RBSP.
- Otherwise (the syntax elements of a slice data partition A RBSP do not indicate the presence of any syntax elements of category 4 in the slice data for a slice), no slice data partition C RBSP shall be present having the same value of slice_id and redundant_pic_cnt as in the slice data partition A RBSP.
- 对于一个 slice 来说，如果 slice 数据块 A RBSP的语法元素表示在 slice 数据中有类别 4 的语法元素，那么就有 slice 数据 C RBSP，并且和 slice 数据块 A RBSP 有相同的 slice_id 和 redundant_pic_cnt。
- 否则（对于一个 slice 来说，如果 slice 数据块 A RBSP的语法元素表示在 slice 数据中没有类别 4 的语法元素），那么就没有和 slice 数据块 A RBSP 有相同slice_id 和 redundant_pic_cnt 的 slice 数据块 C RBSP。

7.4.2.10 RBSP slice trailing bits semantics

7.4.2.10 RBSP slice 后缀比特的语义

cabac_zero_word is a byte-aligned sequence of two bytes equal to 0x0000.

cabac_zero_word 是一个由等于 0x0000 的两个字节组成的字节对齐的序列。

Let NumBytesInVclNALunits be the sum of the values of NumBytesInNALunit for all VCL NAL units of a coded picture.

假设 NumBytesInVclNALunits 是一编码图像的所有 VCL NAL 单元中的 NumBytesInNALunit 值的和。

When entropy_coding_mode_flag is equal to 1, the number of bins resulting from decoding the contents of all VCL NAL units of a coded picture shall not exceed $(32 \div 3) * \text{NumBytesInVclNALunits} + 96 * \text{PicSizeInMbs}$.

当 entropy_coding_mode_flag 等于 1 时，那么解码一编码图像的所有 VCL NAL 单元内容得到的二进制数的数目不能超过 $(32 \div 3) * \text{NumBytesInVclNALunits} + 96 * \text{PicSizeInMbs}$。（不知道这是为什么）

NOTE – The constraint on the maximum number of bins resulting from decoding the contents of the slice layer NAL units can be met by inserting a number of cabac_zero_word syntax elements to increase the value of NumBytesInVclNALunits. Each cabac_zero_word is represented in a NAL unit by the three-byte sequence 0x000003 (as a result of the constraints on NAL unit contents that result in requiring inclusion of an emulation_prevention_three_byte for each cabac_zero_word).

注 — 为了满足对从解码 slice 层 NAL 单元得到的最大二进制数目的限制，可以通过插入一定数量的 cabac_zero_word 语法语法来增加 NumBytesInVclNALunits 的值。每个 cabac_zero_word 在 NAL 单元中被表示成三字节的 0x000003。

7.4.2.11 RBSP trailing bits semantics

7.4.2.11 RBSP 后缀比特的语义

rbsp_stop_one_bit is a single bit equal to 1.

rbsp_stop_one_bit 是一个等于1 的比特位。

rbsp_alignment_zero_bit is a single bit equal to 0.

rbsp_alignment_zero_bit 是一个等于 0 的比特位。

7.4.3 Slice header semantics

7.4.3 slice 头语义

When present, the value of the slice header syntax elements pic_parameter_set_id, frame_num, field_pic_flag, bottom_field_flag, idr_pic_id, pic_order_cnt_lsb, delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], delta_pic_order_cnt[1], sp_for_switch_flag, and slice_group_change_cycle shall be the same in all slice headers of a coded picture.

目前，slice 头语法元素 pic_parameter_set_id，frame_num，field_pic_flag，bottom_field_flag，idr_pic_id，pic_order_cnt_lsb，delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], delta_pic_order_cnt[1], sp_for_switch_flag, 和 slice_group_change_cycle 在一编码图像的所有 slice 头中都是相同的。

first_mb_in_slice specifies the address of the first macroblock in the slice. When arbitrary slice order is not allowed as specified in Annex A, the value of first_mb_in_slice shall not be less than the value of first_mb_in_slice for any other slice of the current picture that precedes the current slice in decoding order.

first_mb_in_slice 表示了 slice 中第一个宏块的地址。当不允许使用附录 A 中的 ASO（arbitrary slice order）时，那么 first_mb_in_slice 的值不应该小于按照解码顺序在当前 slice 之前的 slice 中的 first_mb_in_slice 值。

The first macroblock address of the slice is derived as follows.

Slice 中第一个宏块的地址计算如下：

– If MbaffFrameFlag is equal to 0, first_mb_in_slice is the macroblock address of the first macroblock in the slice, and first_mb_in_slice shall be in the range of 0 to PicSizeInMbs - 1, inclusive.

– Otherwise (MbaffFrameFlag is equal to 1), first_mb_in_slice * 2 is the macroblock address of the first macroblock in the slice, which is the top macroblock of the first macroblock pair in the slice, and first_mb_in_slice shall be in the range of 0 to PicSizeInMbs / 2 - 1, inclusive.

– 如果 MbaffFrameFlag 等于 0，那么 first_mb_in_slice 是 slice 中第一个宏块的地址，first_mb_in_slice 值的范围为 0 到 PicSizeInMbs – 1，包括 0 和 PicSizeInMbs – 1 这两个数。

– 否则（MbaffFrameFlag 等于 1），first_mb_in_slice * 2 是 slice 中第一个宏块的地址，它是 slice 中第一个宏块对的顶场宏块的地址，first_mb_in_slice 值的范围是 0 到 PicSizeInMbs / 2 – 1，包括这两个数。

slice_type specifies the coding type of the slice according to Table 7-3.

slice_type 表示了 slice 的编码类型，如表 Table 7-3 所示。

Table 7-3 – Name association to slice_type

slice_type	Name of slice_type
0	P (P slice)
1	B (B slice)
2	I (I slice)
3	SP (SP slice)
4	SI (SI slice)
5	P (P slice)
6	B (B slice)
7	I (I slice)
8	SP (SP slice)
9	SI (SI slice)

slice_type values in the range 5..9 specify, in addition to the coding type of the current slice, that all other slices of the current coded picture shall have a value of slice_type equal to the current value of slice_type or equal to the current value of slice_type – 5.

范围在 5..9 的slice_type 值表示了，除了当前 slice 的编码类型之外，当前编码图像的其他 slice 应该有和当前 slice_type 的值相等的 slice_type 或者 等于 slice_type 的当前值减去 5。（这样做的目的什么？）

When nal_unit_type is equal to 5 (IDR picture), slice_type shall be equal to 2, 4, 7, or 9.

当 nal_unit_type 等于 5 时（IDR 图像），slice_type 的值应该等于 2、4、7 或者 9。

pic_parameter_set_id specifies the picture parameter set in use. All slices belonging to a picture shall have the same value of pic_parameter_set_id. The value of pic_parameter_set_id shall be in the range of 0 to 255, inclusive.

pic_parameter_set_id 表示了使用的图像参数集。属于同一图像的所有 slice 应该有相同的 pic_parameter_set_id。pic_parameter_set_id 值的范围是 0 到 255，包括 0 和 255。

frame_num is used as a unique identifier for each short-term reference frame and shall be represented by log2_max_frame_num_minus4 + 4 bits in the bitstream. frame_num is constrained as follows:

frame_num 是短期参考图像的唯一标识，在码流中表示成 log2_max_frame_num_minus4 + 4 比特位。对 frame_num 的限制如下：

The variable PrevRefFrameNum is derived as follows.

变量 PrevRefFrameNum 的计算如下：

- If the current picture is an IDR picture, PrevRefFrameNum is set equal to 0.
- Otherwise (the current picture is not an IDR picture), PrevRefFrameNum is set equal to the value of frame_num for the previous access unit in decoding order that contains a reference picture.
- 如果当前图像是一个 IDR 图像，那么 PrevRefFrameNum 被设置成 0。
- 否则（当前图像不是 IDR 图像），那么 PrevRefFrameNum 被设置成等于按照解码顺序包含参考图像的前一访问单元的 frame_num 的值。

The value of frame_num is constrained as follows.

frame_num 的值限制如下：

- If the current picture is an IDR picture, frame_num shall be equal to 0.
- 如果当前图像是 IDR 图像，那么 frame_num 的值被设置成 0。
- Otherwise (the current picture is not an IDR picture), referring to the primary coded picture in the previous access unit in decoding order that contains a reference picture as the preceding reference picture, the value of frame_num for the current picture shall not be equal to PrevRefFrameNum unless all of the following three conditions are true.
- 否则（当前图像不是 IDR 图像），将按照解码顺序包含参考图像的前一个访问单元中的基本编码图像作为前一参考图像，当前图像的 frame_num 值不等于 PrevRefFrameNum，除非下面的三个条件都成立。
  - the current picture and the preceding reference picture belong to consecutive access units in decoding order
  - 当前图像和前面的参考图像在解码顺序上属于连续的访问单元。
  - the current picture and the preceding reference picture are reference fields having opposite parity
  - 当前图像和前面的参考图像都是参考场并且是一互补场对的两个对应场（一个为底场，一个为顶场）
  - one or more of the following conditions is true
  - 满足一个或者多个下面的条件
    – the preceding reference picture is an IDR picture
    – the preceding reference picture includes a memory_management_control_operation syntax element equal to 5

      NOTE – If the preceding reference picture includes a memory_management_control_operation syntax element equal to 5, PrevRefFrameNum is equal to 0.
    – there is a primary coded picture that precedes the preceding reference picture and the primary coded picture that precedes the preceding reference picture does not have frame_num equal to PrevRefFrameNum
    – there is a primary coded picture that precedes the preceding reference picture and the primary coded picture that precedes the preceding reference picture is not a reference picture
    － 前面的参考图像是 IDR 图像
    － 前面的参考图像的语法元素 memory_management_control_operation 等于 5

      注 － 如果前面参考图像的 memory_management_control_operation 等于 5，那么 PrevRefFrameNum 等于 0
    － 在前面的参考图像之前有一个基本编码图像，并且这个基本编码图像的 frame_num 值不等于 PrevRefFrameNum。
    － 在前面的参考图像之前有一个基本编码图像，并且这个基本编码图像不是一个参考图像。

When gaps_in_frame_num_value_allowed_flag is equal to 0 and frame_num is not equal to PrevRefFrameNum, frame_num shall be equal to (PrevRefFrameNum + 1) % MaxFrameNum.

当 gaps_in_frame_num_value_allowed_flag 等于 0 并且 frame_num 不等于 PrevRefFrameNum，那么 frame_num 的值应该等于 (PrevRefFrameNum + 1) % MaxFrameNum。

When the value of frame_num is not equal to PrevRefFrameNum, there shall not be any previous field or frame in decoding order that is currently marked as "used for short-term reference" that has a value of frame_num equal to any value taken on by the variable UnusedShortTermFrameNum in the following:

当 frame_num 的值不等于 PrevRefFrameNum 时，那么按照解码顺序，前面应该没有标识为“用作短期参考”的帧或者场的 frame_num 值等于下面变量 UnusedShortTermFrameNum 的值：

UnusedShortTermFrameNum = (PrevRefFrameNum + 1) % MaxFrameNum
while(UnusedShortTermFrameNum != frame_num) (7-10)
UnusedShortTermFrameNum = (UnusedShortTermFrameNum + 1) % MaxFrameNum

A picture including a memory_management_control_operation equal to 5 shall have frame_num constraints as described above, however, after the decoding of the current picture and the processing of the memory management control operations, shall be inferred to have had frame_num equal to 0 for all subsequent use in the decoding process.

对 memory_management_control_operation 等于 5 的图像的 frame_num 的限制如上面的描述，但是解码完当前图像和缓存管理控制处理之后，对于接下来的解码过程来说，它的 frame_num 值应该等于 0。

NOTE – When the primary coded picture is not an IDR picture and does not contain memory_management_control_operation syntax element equal to 5, the value of frame_num of a corresponding redundant coded picture is the same as the value of frame_num in the primary coded picture. Alternatively, the redundant coded picture includes a memory_management_control_operation syntax element equal to 5 and the corresponding primary coded picture is an IDR picture.

注 – 当基本编码图像不是 IDR 图像并且 memory_management_control_operation 也不等于 5，那么对应冗余编码图像的 frame_num 值和基本编码图像的 frame_num 值是相同的。换句话说，冗余编码图像的 memory_management_control_operation 语法元素等于 5 和基本编码图像是 IDR 图像。

field_pic_flag equal to 1 specifies that the slice is a slice of a coded field. field_pic_flag equal to 0 specifies that the slice is a slice of a coded frame. When field_pic_flag is not present it shall be inferred to be equal to 0.

field_pic_flag 等于 1 表示了 slice 是一编码场的 slice，field_pic_flag 等于 0 表示了 slice 是一编码帧的 slice，如果码流中没有 field_pic_flag 语法元素时，它的值被设置成 0。

The variable MbaffFrameFlag is derived as follows.

变量 MbaffFrameFlag 的计算如下：

MbaffFrameFlag = (mb_adaptive_frame_field_flag && !field_pic_flag) (7-11)

The variable for the picture height in units of macroblocks is derived as follows

以宏块为单位的图像高度变量的计算如下：

PicHeightInMbs = FrameHeightInMbs / (1 + field_pic_flag) (7-12)

The variable for picture height for the luma component is derived as follows

亮度分量图像高度的计算如下：

$\text{PicHeightInSamples}_L = \text{PicHeightInMbs} * 16$ (7-13)

The variable for picture height for the chroma component is derived as follows

色差分量图像高度的计算如下：

$\text{PicHeightInSamples}_C = \text{PicHeightInMbs} * 8$ (7-14)

The variable PicSizeInMbs for the current picture is derived according to:

当前图像的变量 PicSizeInMbs 的计算如下：

PicSizeInMbs = PicWidthInMbs * PicHeightInMbs (7-15)

The variable MaxPicNum is derived as follows.

变量 MaxPicNum 的计算如下：

- If field_pic_flag is equal to 0, MaxPicNum is set equal to MaxFrameNum.
- Otherwise (field_pic_flag is equal to 1), MaxPicNum is set equal to 2*MaxFrameNum.
- 如果 field_pic_flag 等于 0，MaxPicNum 被设置成 MaxFrameNum。
- 否则（field_pic_flag 等于 1），MaxPicNum 被设置成 2*MaxFrameNum。

The variable CurrPicNum is derived as follows.

变量 CurrPicNum 的计算如下：

- If field_pic_flag is equal to 0, CurrPicNum is set equal to frame_num.
- Otherwise (field_pic_flag is equal to 1), CurrPicNum is set equal to 2 * frame_num + 1.
- 如果 field_pic_flag 等于 0，CurrPicNum 被设置成 frame_num；
- 否则（field_pic_flag 等于 1），CurrPicNum 被设置成 2 * frame_num + 1；

bottom_field_flag equal to 1 specifies that the slice is part of a coded bottom field. bottom_field_flag equal to 0 specifies that the picture is a coded top field. When this syntax element is not present for the current slice, it shall be inferred to be equal to 0.

bottom_field_flag 等于 1 表示当前 slice 是一编码底场的 slice，bottom_field_flag 等于 0 表示当前图像是一编码顶场图像。对于当前的 slice，如果没有出现这个语法元素，它将被设置成 0。

idr_pic_id identifies an IDR picture. The values of idr_pic_id in all the slices of an IDR picture shall remain unchanged. When two consecutive access units in decoding order are both IDR access units, the value of idr_pic_id in the slices of the first such IDR access unit shall differ from the idr_pic_id in the second such IDR access unit. The value of idr_pic_id shall be in the range of 0 to 65535, inclusive.

idr_pic_id 表示一个 IDR 图像。IDR 图像中所有 slice 的 idr_pic_id 将保持不变。当按照解码顺序两个连续的访问单元都是 IDR 访问单元，那么第一个 IDR 访问单元的 idr_pic_id 值和第二个访问单元的 idr_pic_id 值是不相同的。idr_pic_id 值的范围是 0 到 65535，包括这两个数。

pic_order_cnt_lsb specifies the picture order count coded in modulo MaxPicOrderCntLsb arithmetic for the top field of a coded frame or for a coded field. An IDR picture shall have pic_order_cnt_lsb equal to 0. The size of the pic_order_cnt_lsb variable is log2_max_pic_order_cnt_lsb_minus4 + 4 bits. The value of the pic_order_cnt_lsb shall be in the range of 0 to MaxPicOrderCntLsb – 1, inclusive.

pic_order_cnt_lsb 表示了对于一编码帧的顶场或者编码场的图像顺序数与 MaxPicOrderCntLsb 的算术模。一 IDR 图像的 pic_order_cnt_lsb 等于 0，pic_order_cnt_lsb 变量的大小是 log2_max_pic_order_cnt_lsb_minus4 + 4 比特位。pic_order_cnt_lsb 值的范围是 0 到 MaxPicOrderCntLsb – 1，包括这两个数。

delta_pic_order_cnt_bottom specifies the picture order count difference between the bottom field and the top field of a coded frame. The value of delta_pic_order_cnt_bottom shall be in the range of -2^{31} to $2^{31} - 1$, inclusive. When this syntax element is not present in the bitstream for the current slice, it shall be inferred to be equal to 0.

delta_pic_order_cnt_bottom 表示了一编码帧的底场和顶场之间的图像顺序数的差。delta_pic_order_cnt_bottom 值的范围是 -2^{31} 到 $2^{31} - 1$，包括这两个数。对于当前的 slice，如果码流中没有这个语法元素，那么它的值将被设置成 0。

delta_pic_order_cnt[0 **]** specifies the picture order count difference from the expected picture order count for the top field of a coded frame or for a coded field as specified in subclause 8.2.1. The value of delta_pic_order_cnt[0] shall be in the range of -2^{31} to $2^{31} - 1$, inclusive. When this syntax element is not present in the bitstream for the current slice, it shall be inferred to be equal to 0.

delta_pic_order_cnt[0 **]** 表示了编码帧的顶场或者一编码场的期望图像顺序数和实际图像顺序数的差值，如 8.2.1 小节所示。delta_pic_order_cnt[0] 值的范围是 -2^{31} 到 $2^{31} - 1$，包括这两个数。对于当前的 slice，如果码流中没有这个语法元素，那么它的值将被设置成 0。

delta_pic_order_cnt[1 **]** specifies the picture order count difference from the expected picture order count for the bottom field of a coded frame specified in subclause 8.2.1. The value of delta_pic_order_cnt[1] shall be in the range of -2^{31} to $2^{31} - 1$, inclusive. When this syntax element is not present in the bitstream for the current slice, it shall be inferred to be equal to 0.

delta_pic_order_cnt[1 **]** 表示了编码帧的底场或者一编码场的期望图像顺序数和实际图像顺序数的差值，如 8.2.1 小节所示。delta_pic_order_cnt[1] 值的范围是 -2^{31} 到 $2^{31} - 1$，包括这两个数。对于当前的 slice，如果码流中没有这个语法元素，那么它的值将被设置成 0。

redundant_pic_cnt shall be equal to 0 for slices and slice data partitions belonging to the primary coded picture. The value of redundant_pic_cnt shall be greater than 0 for coded slices or coded slice data partitions of a redundant coded picture. When redundant_pic_cnt is not present in the bitstream, its value shall be inferred to be equal to 0. The value of redundant_pic_cnt shall be in the range of 0 to 127, inclusive.

redundant_pic_cnt 对于基本编码图像的 slice 或 slice 数据块来说应该等于 0。对于冗余编码图像的 slice 或者 slice 数据块来说，redundant_pic_cnt 的值应该大于 0，如果在码流中没有这个语法元素，那么它的值被设置成 0。redundant_pic_cnt 值的范围是 0 到 127，包括 0 和 127。

NOTE - There should be no noticeable difference between the co-located areas of the decoded primary picture and any decoded redundant pictures.

注 － 在解码基本图像和解码冗余图像的对应位置的图像之间应该没有差别。

The value of pic_parameter_set_id in a coded slice or coded slice data partition of a redundant coded picture shall be such that the value of pic_order_present_flag in the picture parameter set in use in a redundant coded picture is equal to the value of pic_order_present_flag in the picture parameter set in use in the corresponding primary coded picture.

冗余编码图像的编码 slice 或者编码 slice 数据块的 pic_parameter_set_id 值应该使冗余编码图像的图像参数集中的 pic_order_present_flag 值和对应的基本编码图像的图像参数集中的 pic_order_present_flag 值相等。

When present in the primary coded picture and any redundant coded picture, the following syntax elements shall have the same value: field_pic_flag, bottom_field_flag, idr_pic_id, pic_order_cnt_lsb, pic_order_cnt_lsb, delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], and delta_pic_order_cnt[1].

在冗余编码图像和基本编码图像中，下面的语法元素应该有相同的值：field_pic_flag, bottom_field_flag, idr_pic_id, pic_order_cnt_lsb, pic_order_cnt_lsb, delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], 和 delta_pic_order_cnt[1]。

When the value of nal_ref_idc in one VCL NAL unit of an access unit is equal to 0, the value of nal_ref_idc in all other VCL NAL units of the same access unit shall be equal to 0.

当一个访问单元中的一个 VCL NAL 单元的 nal_ref_idc 值等于 0，那么在相同访问单元中的其它 VCL NAL 单元的 nal_ref_idc 也应该等于 0。

NOTE – The above constraint also has the following implications. If the value of nal_ref_idc for the VCL NAL units of the primary coded picture is equal to 0, the value of nal_ref_idc for the VCL NAL units of any corresponding redundant coded picture are equal to 0; otherwise (the value of nal_ref_idc for the VCL NAL units of the primary coded picture is greater than 0), the value of nal_ref_idc for the VCL NAL units of any corresponding redundant coded picture are also greater than 0.

注 － 上面的限制主要有下面的一些含义：如果基本编码图像的 VCL NAL 单元的 nal_ref_idc 值等于 0，那么对应的冗余编码图像的 VCL NAL 单元的 nal_ref_idc 值也等于 0；否则（基本编码图像的 VCL NAL 单元的 nal_ref_idc 值大于 0），对应的冗余编码图像的 VCL NAL 单元的 nal_ref_idc 值也大于 0。

The marking status of reference pictures and the value of frame_num after the decoded reference picture marking process as specified in subclause 8.2.5 is invoked for the primary coded picture or any redundant coded picture of the same access unit shall be identical regardless whether the primary coded picture or any redundant coded picture (instead of the primary coded picture) of the access unit would be decoded.

对于相同访问单元的基本编码图像或者冗余编码图像来说，参考帧的标识状态和解码参考图像标识过程之后的 frame_num 值应该是相同的，如 8.2.5 小节所示，不管访问单元中的基本编码图像或者冗余编码图像（用于替代基本编码图像）是否解码。

NOTE – The above constraint also has the following implications.

注 － 上面的限制有下面的一些含义：

If a primary coded picture is not an IDR picture, the contents of the dec_ref_pic_marking( ) syntax structure must be identical in all slice headers of the primary coded picture and all redundant coded pictures corresponding to the primary coded picture.

Otherwise (a primary coded picture is an IDR picture), the following applies.

如果基本编码图像不是 IDR 图像，那么在基本编码图像和对应的冗余编码图像中所有 slice 头的 dec_ref_pic_marking( ) 语法结构的内容必须相同。否则（基本编码图像是 IDR 图像），使用下面的过程：

If a redundant picture corresponding to the primary coded picture is not an IDR pictures, all slice headers of the redundant picture must contain a dec_ref_pic_marking syntax( ) structure including a memory_management_control_operation syntax element equal to 5.

如果对应于基本编码图像的冗余编码图像不是 IDR 图像，那么冗余图像的所有 slice 头必须包含 dec_ref_pic_marking 语法结构，并且语法结构 dec_ref_pic_marking 中的 memory_management_control_operation 语法元素等于 5。

Otherwise (the redundant coded picture is an IDR picture) the following applies.

否则（冗余编码图像是 IDR 图像），使用下面的过程：

The contents of the dec_ref_pic_marking( ) syntax structure must be identical in all slice headers of the primary coded picture and the redundant coded picture corresponding to the primary coded picture.

在基本编码图像和对应的冗余编码图像中所有 slice 头的 dec_ref_pic_marking( ) 语法结构的内容必须完全相同。

If the value of long_term_reference_flag in the primary coded picture is equal to 0, the dec_ref_pic_marking syntax structure of the redundant coded picture must not include a memory_management_control_operation syntax element equal to 6.

如果在基本编码图像中 long_term_reference_flag 的值等于 0，那么冗余编码图像中dec_ref_pic_marking 语法结构的 memory_management_control_operation 语法元素不等于 6。

Otherwise (the value of long_term_reference_flag in the primary coded picture is equal to 1), the value of the MaxLongTermFrameIdx variable before decoding the primary coded picture must be equal to 0 and the dec_ref_pic_marking syntax structure of the redundant coded picture must include memory_management_control_operation syntax elements equal to 5, 4, and 6 in decoding order, and the value of max_long_term_frame_idx_plus1 must be equal to 1, and the value of long_term_frame_idx must be equal to 0.

否则（基本编码图像中 long_term_reference_flag 的值等于 1），解码基本编码图像之前变量 MaxLongTermFrameIdx 的值必须等于 0 并且按照解码顺序冗余编码图像的 dec_ref_pic_marking 语法结构中必须包含 memory_management_control_operation 等于 5 、4 和 6 的语法元素，max_long_term_frame_idx_plus1 的值必须等于 1，long_term_frame_idx 的值必须等于 0。

There is no required decoding process for a coded slice or coded slice data partition of a redundant coded picture. When the redundant_pic_cnt in the slice header of a coded slice is greater than 0, the decoder may discard the coded slice. However, a coded slice or coded slice data partition of any redundant coded picture shall obey the same constraints as a coded slice or coded slice data partition of a primary picture.

对于冗余编码图像中的编码 slice 和编码 slice 数据块来说，没有解码的需求。当在编码 slice 的头中的 redundant_pic_cnt 大于 0，解码器可以忽略这个编码 slice。但是，任何冗余编码图像的编码 slice 或者 编码 slice 数据块应该遵循与基本编码图像中编码 slice 和 编码 slice 数据块相同的限制。

NOTE – When some of the samples in the decoded primary picture cannot be correctly decoded due to errors or losses in transmission of the sequence and the coded redundant slice can be correctly decoded, the decoder should replace the samples of the decoded primary picture with the corresponding samples of the decoded redundant slice. When more than one redundant slice covers the relevant region of the primary picture, the redundant slice having the lowest value of redundant_pic_cnt should be used.

注 – 当由于传输中的错误或者丢包导致基本解码图像中的一些象素不能正确解码二而编码冗余 slice 可以正确的解码时，那么解码器应该用解码冗余 slice 的正确象素值来取代解码基本图像中的象素值。当多个冗余 slice 覆盖基本图像的相关区域时，就使用 redundant_pic_cnt 值最低的冗余 slice。

Redundant slices and slice data partitions having the same value of redundant_pic_cnt belong to the same redundant picture. Decoded slices within the same redundant picture need not cover the entire picture area and shall not overlap.

具有相同 redundant_pic_cnt 值的冗余 slice 和 slice 数据块属于相同的冗余图像。在相同的冗余图像中解码 slice 不需要覆盖整个图像区域和重叠。

direct_spatial_mv_pred_flag specifies the method used in the decoding process to derive motion vectors and reference indices for inter prediction.

direct_spatial_mv_pred_flag 表示了帧间预测过程中计算运动矢量和参考索引时使用的方法。

If direct_spatial_mv_pred_flag is equal to 1, the derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8 in subclause 8.4.1.2 shall use spatial direct mode prediction as specified in subclause 8.4.1.2.2.

如果 direct_spatial_mv_pred_flag 等于 1，那么对于 B_Skip, B_Direct_16x16, 和 B_Direct_8x8 ，运动矢量的计算过程使用空间直接预测模式，如 8.4.1.2.2 小节所示。

Otherwise (direct_spatial_mv_pred_flag is equal to 0), the derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8 in subclause 8.4.1.2 shall use temporal direct mode prediction as specified in subclause 8.4.1.2.3.

否则（direct_spatial_mv_pred_flag 等于 0），那么对于 B_Skip, B_Direct_16x16, 和 B_Direct_8x8 ，运动矢量的计算过程使用时间直接预测模式，如 8.4.1.2.3 小节所示。

num_ref_idx_active_override_flag equal to 0 specifies that the values of the syntax elements num_ref_idx_l0_active_minus1 and num_ref_idx_l1_active_minus1 specified in the referred picture parameter set are in effect. num_ref_idx_active_override_flag equal to 1 specifies that the num_ref_idx_l0_active_minus1 and num_ref_idx_l1_active_minus1 specified in the referred picture parameter set are overridden for the current slice (and only for the current slice) by the following values in the slice header.

num_ref_idx_active_override_flag 等于 0 表示了参考的图像参数集中 num_ref_idx_l0_active_minus1 和 num_ref_idx_l1_active_minus1 语法元素的值有效。num_ref_idx_active_override_flag 等于 1 表示了当前 slice 参考的图像参数集中 num_ref_idx_l0_active_minus1 和 num_ref_idx_l1_active_minus1 语法元素的值将被slice 头中的下面的值替换（这些替换的值仅仅用于当前的 slice）。

When the current slice is a P, SP, or B slice and field_pic_flag is equal to 0 and the value of num_ref_idx_l0_active_minus1 in the picture parameter set exceeds 15, num_ref_idx_active_override_flag shall be equal to 1.

当当前 slice 是 P, SP, 或 B slice 和 field_pic_flag 等于 0 和 图像参数集中 num_ref_idx_l0_active_minus1 的值大于 15 时，num_ref_idx_active_override_flag 的值应该等于 1。

When the current slice is a B slice and field_pic_flag is equal to 0 and the value of num_ref_idx_l1_active_minus1 in the picture parameter set exceeds 15, num_ref_idx_active_override_flag shall be equal to 1.

当当前的 slice 是 B slice 和 field_pic_flag 等于 0 和 图像参数集中的 num_ref_idx_l1_active_minus1 值大于 15，那么 num_ref_idx_active_override_flag 的值应该等于 1。

num_ref_idx_l0_active_minus1 specifies the maximum reference index for reference picture list 0 that shall be used to decode the slice.

num_ref_idx_l0_active_minus1 表示了用于解码 slice 的参考图像列表 0 中的最大参考索引值。

The range of num_ref_idx_l0_active_minus1 is specified as follows.

num_ref_idx_l0_active_minus1 值的范围说明如下：

– If field_pic_flag is equal to 0, num_ref_idx_l0_active_minus1 shall be in the range of 0 to 15, inclusive. When MbaffFrameFlag is equal to 1, num_ref_idx_l0_active_minus1 is the maximum index value for the decoding of frame macroblocks and 2 * num_ref_idx_l0_active_minus1 + 1 is the maximum index value for the decoding of field macroblocks.

– Otherwise (field_pic_flag is equal to 1), num_ref_idx_l0_active_minus1 shall be in the range of 0 to 31, inclusive.

– 如果 field_pic_flag 等于 0，num_ref_idx_l0_active_minus1 的值应该在 0 到 15 之间，包括 0 和 15。当 MbaffFrameFlag 等于 1 时，num_ref_idx_l0_active_minus1 是解码帧宏块的最大索引值，2 * num_ref_idx_l0_active_minus1 + 1 是解码场宏块的最大索引值。

– 否则（field_pic_flag 等于 1），num_ref_idx_l0_active_minus1 的值应该在 0 到 31 之间，包括 0 和 31。

num_ref_idx_l1_active_minus1 has the same semantics as num_ref_idx_l0_active_minus1 with l0 and list 0 replaced by l1 and list 1, respectively.

num_ref_idx_l1_active_minus1 的语义和 num_ref_idx_l0_active_minus1 相同，l0 和 列表 0 分别被替换成 l1 和 列表 1。

cabac_init_idc specifies the index for determining the initialisation table used in the initialisation process for context variables. The value of cabac_init_idc shall be in the range of 0 to 2, inclusive.

cabac_init_idc 表示了内容变量初始化过程中使用的初始化表的索引。cabac_init_idc 值应该在 0 到 2 之间，包括 0 和 2 这两个数。

slice_qp_delta specifies the initial value of QP_Y to be used for all the macroblocks in the slice until modified by the value of mb_qp_delta in the macroblock layer. The initial QP_Y quantisation parameter for the slice is computed as:

slice_qp_delta 表示了 slice 中所有宏块的 QP_Y 的初始值，一直到被宏块层的 mb_qp_delta 值修改为止。Slice 层的量化参数 QP_Y 的初始值的计算如下：

$$SliceQP_Y = 26 + pic\_init\_qp\_minus26 + slice\_qp\_delta \quad (7\text{-}16)$$

The value of slice_qp_delta shall be limited such that QP_Y is in the range of 0 to 51, inclusive.

slice_qp_delta 的值应该能够保证 QP_Y 的值在 0 到 51 之间，包括 0 和 51。

sp_for_switch_flag specifies the decoding process to be used to decode P macroblocks in an SP slice as follows.

sp_for_switch_flag 表示了在 SP slice 中用于解码 P 宏块的解码过程。

- If sp_for_switch_flag is equal to 0, the P macroblocks in the SP slice shall be decoded using the SP decoding process for non-switching pictures as specified in subclause 8.6.1.
- Otherwise (sp_for_switch_flag is equal to 1), the P macroblocks in the SP slice shall be decoded using the SP and SI decoding process for switching pictures as specified in subclause 8.6.2.
- 如果 sp_for_switch_flag 等于 0，那么 SP slice 中 P 宏块的解码过程应该使用非切换图像的 SP 解码过程，如 8.6.1 小节所示。
- 否则（sp_for_switch_flag 等于 1），那么 SP slice 中 P 宏块的解码过程应该使用切换图像的 SP 和 SI 的解码过程，如 8.6.2 小节所示。

slice_qs_delta specifies the value of QS_Y for all the macroblocks in SP and SI slices. The QS_Y quantisation parameter for the slice is computed as:

slice_qs_delta 表示了 SP 和 SI slice 中所有宏块的 QS_Y 的值，slice 的 QS_Y 量化参数的计算如下：

$$QS_Y = 26 + \text{pic\_init\_qs\_minus26} + \text{slice\_qs\_delta} \tag{7-17}$$

The value of slice_qs_delta shall be limited such that QS_Y is in the range of 0 to 51, inclusive. This value of QS_Y is used for the decoding of all macroblocks in SI slices with mb_type equal to SI and all macroblocks in SP slices with prediction mode equal to inter.

slice_qs_delta 的值应该能够保证 QS_Y 的值在 0 到 51 之间，包括 0 和 51。QS_Y 的值被用来解码所有宏块类型为 SI 的宏块和使用帧间预测模式的 SP slice 中的所有宏块。

disable_deblocking_filter_idc specifies whether the operation of the deblocking filter shall be disabled across some block edges of the slice and specifies for which edges the filtering is disabled. When disable_deblocking_filter_idc is not present in the slice header, the value of disable_deblocking_filter_idc shall be inferred to be equal to 0.

disable_deblocking_filter_idc 表示了对 slice 的一些块边缘是否禁用环路滤波操作和对哪些块边缘禁用环路滤波操作。当在 slice 头中没有 disable_deblocking_filter_idc 时，disable_deblocking_filter_idc 的值被设置成等于 0。

The value of disable_deblocking_filter_idc shall be in the range of 0 to 2, inclusive.

disable_deblocking_filter_idc 的值应该在 0 和 2 之间，包括 0 和 2。

slice_alpha_c0_offset_div2 specifies the offset used in accessing the α and C0 deblocking filter tables for filtering operations controlled by the macroblocks within the slice. From this value, the offset that shall be applied when addressing these tables shall be computed as:

slice_alpha_c0_offset_div2 表示了访问 α 和 C0 环路滤波表时的偏移量，用于控制 slice 内的宏块滤波操作。当访问这些表时，根据这个值，使用的实际偏移量计算如下：

$$\text{FilterOffsetA} = \text{slice\_alpha\_c0\_offset\_div2} \ll 1 \tag{7-18}$$

The value of slice_alpha_c0_offset_div2 shall be in the range of -6 to +6, inclusive. When slice_alpha_c0_offset_div2 is not present in the slice header, the value of slice_alpha_c0_offset_div2 shall be inferred to be equal to 0.

slice_alpha_c0_offset_div2 值应该在 -6 到 +6 的范围内，包括 -6 和 +6 这两个数。在 slice 中如果没有 slice_alpha_c0_offset_div2 这个语法元素时，那么 slice_alpha_c0_offset_div2 的值被设置成 0。

slice_beta_offset_div2 specifies the offset used in accessing the β deblocking filter table for filtering operations controlled by the macroblocks within the slice. From this value, the offset that is applied when addressing the β table of the deblocking filter shall be computed as:

slice_beta_offset_div2 表示了访问 β 环路滤波表时的偏移量，用于控制 slice 内的宏块滤波操作。当访问 β 环路滤波表时，根据这个值，使用的实际偏移量计算如下：

$$\text{FilterOffsetB} = \text{slice\_beta\_offset\_div2} \ll 1 \tag{7-19}$$

The value of slice_beta_offset_div2 shall be in the range of -6 to +6, inclusive. When slice_beta_offset_div2 is not present in the slice header the value of slice_beta_offset_div2 shall be inferred to be equal to 0.

slice_beta_offset_div2 值应该在 -6 到 +6 的范围内，包括 -6 和 +6 这两个数。在 slice 中如果没有 slice_beta_offset_div2 这个语法元素时，那么 slice_alpha_c0_offset_div2 的值被设置成 0。

slice_group_change_cycle is used to derive the number of slice group map units in slice group 0 when slice_group_map_type is equal to 3, 4, or 5, as specified by

slice_group_change_cycle 用于计算当 slice_group_map_type 等于 3、4 或 5 时在 slice 组 0 中 slice 组映射单元的数目，具体说明如下：

$$\text{MapUnitsInSliceGroup0} = \text{Min}(\text{ slice\_group\_change\_cycle} * \text{SliceGroupChangeRate}, \text{PicSizeInMapUnits }) \tag{7-20}$$

The value of slice_group_change_cycle is represented in the bitstream by the following number of bits

slice_group_change_cycle 值在码流中占用的位数是：

$$\text{Ceil}(\text{ Log2}(\text{ PicSizeInMapUnits} \div \text{SliceGroupChangeRate} + 1\text{)}\text{)} \tag{7-21}$$

The value of slice_group_change_cycle shall be in the range of 0 to Ceil(PicSizeInMapUnits÷SliceGroupChangeRate), inclusive.

slice_group_change_cycle 的值应该在 0 到 Ceil(PicSizeInMapUnits÷SliceGroupChangeRate) 之间，包括这两个数。

7.4.3.1 Reference picture list reordering semantics

7.4.3.1 参考图像列表重排的语义

The syntax elements reordering_of_pic_nums_idc, abs_diff_pic_num_minus1, and long_term_pic_num specify the change from the initial reference picture lists to the reference picture lists to be used for decoding the slice.

语法元素 reordering_of_pic_nums_idc, abs_diff_pic_num_minus1, 和 long_term_pic_num 表示了将初始参考图像列表改变成用于解码 slice 的参考图像列表。

ref_pic_list_reordering_flag_l0 equal to 1 specifies that the syntax element reordering_of_pic_nums_idc is present for specifying reference picture list 0. ref_pic_list_reordering_flag_l0 equal to 0 specifies that this syntax element is not present.

ref_pic_list_reordering_flag_l0 等于 1 表示在码流中有说明参考图像列表 0 的语法元素 reordering_of_pic_nums_idc 。ref_pic_list_reordering_flag_l0 等于 0 表示码流中没有 reordering_of_pic_nums_idc 语法元素。

When ref_pic_list_reordering_flag_l0 is equal to 1, the number of times that reordering_of_pic_nums_idc is not equal to 3 following ref_pic_list_reordering_flag_l0 shall not exceed num_ref_idx_l0_active_minus1 + 1.

当 ref_pic_list_reordering_flag_l0 等于 1 时，在 ref_pic_list_reordering_flag_l0 之后 reordering_of_pic_nums_idc 不等于 3 的次数不应该超过 num_ref_idx_l0_active_minus1 + 1。

When RefPicList0[num_ref_idx_l0_active_minus1] in the initial reference picture list produced as specified in subclause 8.2.4.2 is equal to "no reference picture", ref_pic_list_reordering_flag_l0 shall be equal to 1 and reordering_of_pic_nums_idc shall not be equal to 3 until RefPicList0[num_ref_idx_l0_active_minus1] in the reordered list produced as specified in subclause 8.2.4.3 is not equal to "no reference picture".

当由 8.2.4.2 小节产生的初始参考图像列表中的 RefPicList0[num_ref_idx_l0_active_minus1] 等于“没有参考图像”时，ref_pic_list_reordering_flag_l0 应该等于 1 并且 reordering_of_pic_nums_idc 不等于 3 直到 8.2.4.3 小节产生的重排列表中的 RefPicList0[num_ref_idx_l0_active_minus1] 不等于 “没有参考图像”。

ref_pic_list_reordering_flag_l1 equal to 1 specifies that the syntax element reordering_of_pic_nums_idc is present for specifying reference picture list 1. ref_pic_list_reordering_flag_l1 equal to 0 specifies that this syntax element is not present.

ref_pic_list_reordering_flag_l1 等于 1 表示在码流中有说明参考图像列表 1 的语法元素 reordering_of_pic_nums_idc 。ref_pic_list_reordering_flag_l0 等于 0 表示码流中没有 reordering_of_pic_nums_idc 语法元素。

When ref_pic_list_reordering_flag_l1 is equal to 1, the number of times that reordering_of_pic_nums_idc is not equal to 3 following ref_pic_list_reordering_flag_l1 shall not exceed num_ref_idx_l1_active_minus1 + 1.

当 ref_pic_list_reordering_flag_l1 等于 1 时，在 ref_pic_list_reordering_flag_l1 之后 reordering_of_pic_nums_idc 不等于 3 的次数不应该超过 num_ref_idx_l1_active_minus1 + 1。

When decoding a B slice and RefPicList1[num_ref_idx_l1_active_minus1] in the initial reference picture list produced as specified in subclause 8.2.4.2 is equal to "no reference picture", ref_pic_list_reordering_flag_l1 shall be equal to 1 and reordering_of_pic_nums_idc shall not be equal to 3 until RefPicList1[num_ref_idx_l1_active_minus1] in the reordered list produced as specified in subclause 8.2.4.3 is not equal to "no reference picture".

当解码 B slice 和由 8.2.4.2 小节产生的初始参考图像列表中的 RefPicList1[num_ref_idx_l1_active_minus1] 等于“没有参考图像”时，ref_pic_list_reordering_flag_l1 应该等于 1 并且 reordering_of_pic_nums_idc 不等于 3 直到 8.2.4.3 小节产生的重排列表中的 RefPicList1[num_ref_idx_l1_active_minus1] 不等于 “没有参考图像”。

reordering_of_pic_nums_idc together with abs_diff_pic_num_minus1 or long_term_pic_num specifies which of the reference pictures are re-mapped. The values of reordering_of_pic_nums_idc are specified in Table 7-4. The value of the first reordering_of_pic_nums_idc that follows immediately after ref_pic_list_reordering_flag_l0 or ref_pic_list_reordering_flag_l1 shall not be equal to 3.

reordering_of_pic_nums_idc 和 abs_diff_pic_num_minus1 或者 long_term_pic_num 一起来表示哪个参考图像需要重排序。reordering_of_pic_nums_idc 的值如表 Table 7-4 所示。紧接在 ref_pic_list_reordering_flag_l0 或者 ref_pic_list_reordering_flag_l1 之后的第一个 reordering_of_pic_nums_idc 应该不等于 3。

Table 7-4 – reordering_of_pic_nums_idc operations for reordering of reference picture lists

reordering_of_pic_nums_idc	Reordering specified
0	abs_diff_pic_num_minus1 is present and corresponds to a difference to subtract from a picture number prediction value
1	abs_diff_pic_num_minus1 is present and corresponds to a difference to add to a picture number prediction value
2	long_term_pic_num is present and specifies the long-term picture number for a reference picture
3	End loop for reordering of the initial reference picture list

abs_diff_pic_num_minus1 plus 1 specifies the absolute difference between the picture number of the picture being moved to the current index in the list and the picture number prediction value.

abs_diff_pic_num_minus1 加 1 表示了图像数（picture number）预测值与被移到列表中当前索引的实际图像数之间的绝对差。

The range of abs_diff_pic_num_minus1 is specified as follows.

abs_diff_pic_num_minus1 的范围说明如下：

- If reordering_of_pic_nums_idc is equal to 0, abs_diff_pic_num_minus1 shall be in the range of 0 to MaxPicNum / 2 - 1.
- Otherwise (reordering_of_pic_nums_idc is equal to 1), abs_diff_pic_num_minus1 shall be in the range of 0 to MaxPicNum / 2 - 2.
- 如果 reordering_of_pic_nums_idc 等于 0，abs_diff_pic_num_minus1 的范围是 0 到 MaxPicNum / 2 – 1。
- 否则（reordering_of_pic_nums_idc 等于 1），abs_diff_pic_num_minus1 的范围是 0 到 MaxPicNum / 2 – 2。

The allowed values of abs_diff_pic_num_minus1 are further restricted as specified in subclause 8.2.4.3.1.

abs_diff_pic_num_minus1 的值在 8.2.4.3.1 小节会进一步的受限制。

long_term_pic_num specifies the long-term picture number of the picture being moved to the current index in the list. When decoding a coded frame, long_term_pic_num shall be equal to a LongTermPicNum assigned to one of the reference frames or complementary reference field pair marked as "used for long-term reference". When decoding a coded field, long_term_pic_num shall be equal to a LongTermPicNum assigned to one of the reference fields marked as "used for long-term reference".

long_term_pic_num 表示了将被移到列表中当前索引的图像的长期图像数。当解码一编码帧时，long_term_pic_num 应该等于分配给标识为“用作长期参考”的一参考帧或者一互补参考场对的 LongTermPicNum。当解码一编码场时，long_term_pic_num 应该等于分配给标识为“用作长期参考”的一参考场。

7.4.3.2 Prediction weight table semantics

7.4.3.2 预测加权表语义

luma_log2_weight_denom is the base 2 logarithm of the denominator for all luma weighting factors. The value of luma_log2_weight_denom shall be in the range of 0 to 7, inclusive.

luma_log2_weight_denom 是所有亮度加权系数的分母以 2 为底的对数。luma_log2_weight_denom 值的范围是 0 到 7，包括 0 和 7。

chroma_log2_weight_denom is the base 2 logarithm of the denominator for all chroma weighting factors. The value of chroma_log2_weight_denom shall be in the range of 0 to 7, inclusive.

chroma_log2_weight_denom 是所有色差加权系数的分母以 2 为底的对数。chroma_log2_weight_denom 值的范围是 0 到 7，包括 0 和 7。

luma_weight_l0_flag equal to 1 specifies that weighting factors for the luma component of list 0 prediction are present. luma_weight_l0_flag equal to 0 specifies that these weighting factors are not present.

luma_weight_l0_flag 等于 1 表示有列表 0 预测的亮度分量的加权系数。 luma_weight_l0_flag 等于 0 表示这个系数不存在。

luma_weight_l0[i **]** is the weighting factor applied to the luma prediction value for list 0 prediction using RefPicList0[i]. The value of luma_weight_l0[i] shall be in the range of –128 to 127, inclusive. When luma_weight_l0_flag is equal to 0, luma_weight_l0[i] shall be inferred to be equal to $2^{\text{luma\_log2\_weight\_denom}}$ for RefPicList0[i].

luma_weight_l0[i **]** 表示用于亮度预测值的加权系数，该亮度预测值是使用列表 0 中的 RefPicList0[i] 预测得到的。luma_weight_l0[i] 值的范围是 –128 到 127，包括这两个数。当 luma_weight_l0_flag 等于 0 时，对于 RefPicList0[i]，luma_weight_l0[i] 被设置成 $2^{\text{luma\_log2\_weight\_denom}}$。

luma_offset_l0[i **]** is the additive offset applied to the luma prediction value for list 0 prediction using RefPicList0[i]. The value of luma_offset_l0[i] shall be in the range of –128 to 127, inclusive. When luma_weight_l0_flag is equal to 0, luma_offset_l0[i] shall be inferred as equal to 0 for RefPicList0[i].

luma_offset_l0[i **]** 是用于亮度预测值的加性偏移，该亮度预测值是使用列表 0 中的 RefPicList0[i] 预测得到的。luma_offset_l0[i] 值的范围是 –128 到 127，当 luma_weight_l0_flag 等于 0 时，对于 RefPicList0[i]，luma_offset_l0[i] 被设置成 0。

chroma_weight_l0_flag equal to 1 specifies that weighting factors for the chroma prediction values of list 0 prediction are present. chroma_weight_l0_flag equal to 0 specifies that these weighting factors are not present.

chroma_weight_l0_flag 等于 1 表示有列表 0 预测的色差分量的加权系数。 chroma_weight_l0_flag 等于 0 表示这个系数不存在。

chroma_weight_l0[i **][** j **]** is the weighting factor applied to the chroma prediction values for list 0 prediction using RefPicList0[i] with j equal to 0 for Cb and j equal to 1 for Cr. The value of chroma_weight_l0[i][j] shall be in the range of –128 to 127, inclusive. When chroma_weight_l0_flag is equal to 0**,** chroma_weight_l0[i] shall be inferred to be equal to $2^{\text{chroma\_log2\_weight\_denom}}$ for RefPicList0[i].

chroma_weight_l0[i **][** j **]** 是用于色差预测值的加权系数，这个色差预测值是使用列表 0 中的 RefPicList0[i] 得到的，对于 Cb，j 等于 0，对于 Cr，j 等于 1，chroma_weight_l0[i][j] 值的范围应该在 –128 到127 之间，包括 –128 和127 这两个数。当 chroma_weight_l0_flag 等于 0 时，对于 RefPicList0[i]， chroma_weight_l0 [i] 将被设置成 $2^{\text{chroma\_log2\_weight\_denom}}$。

chroma_offset_l0[i **][** j **]** is the additive offset applied to the chroma prediction values for list 0 prediction using RefPicList0[i] with j equal to 0 for Cb and j equal to 1 for Cr. The value of chroma_offset_l0[i][j] shall be in the range of -128 to 127, inclusive. When chroma_weight_l0_flag is equal to 0**,** chroma_offset_l0[i] shall be inferred to be equal to 0 for RefPicList0[i].

chroma_offset_l0[i **][** j **]** 是用于色差预测值的加性偏移，这个色差预测值是使用列表 0 中的 RefPicList0[i] 得到的，对于 Cb，j 等于 0，对于 Cr，j 等于 1，chroma_offset_l0[i][j] 值的范围是 -128 到 127，包括这两个数。当 chroma_weight_l0_flag 等于 0 时，对于 RefPicList0[i]，chroma_offset_l0[i] 的值将被设置成 0。

luma_weight_l1_flag, luma_weight_l1, **luma_offset_l1**, **chroma_weight_l1_flag**, **chroma_weight_l1**, **chroma_offset_l1** have the same semantics as luma_weight_l0_flag, luma_weight_l0, luma_offset_l0, chroma_weight_l0_flag, chroma_weight_l0, chroma_offset_l0, respectively, with l0, list 0, and List0 replaced by l1, list 1, and List1, respectively.

luma_weight_l1_flag, luma_weight_l1, **luma_offset_l1**, **chroma_weight_l1_flag**, **chroma_weight_l1**, **chroma_offset_l1** 分别和 luma_weight_l0_flag, luma_weight_l0, luma_offset_l0, chroma_weight_l0_flag, chroma_weight_l0, chroma_offset_l0 有相同的语义，由 l1, 列表 1, 和 List1 分别替代 l0, 列表 0, 和 List0。

7.4.3.3 Decoded reference picture marking semantics

7.4.3.3 解码参考图像标识语义

The syntax elements no_output_of_prior_pics_flag, long_term_reference_flag, adaptive_ref_pic_marking_mode_flag, memory_management_control_operation, difference_of_pic_nums_minus1, long_term_frame_idx, long_term_pic_num, and max_long_term_frame_idx_plus1 specify marking of the reference pictures.

语法元素 no_output_of_prior_pics_flag, long_term_reference_flag, adaptive_ref_pic_marking_mode_flag, memory_management_control_operation, difference_of_pic_nums_minus1, long_term_frame_idx, long_term_pic_num, 和 max_long_term_frame_idx_plus1 表示了参考图像的标识。

The marking of a reference picture can be "unused for reference", "used for short-term reference", or "used for long-term reference", but only one of these three. When a reference picture is referred to have the marking "used for reference" this collectively refers to the picture being marked as "used for short-term reference" or "used for long-term reference", but not both.

参考图像的标识可能是“不用作参考”、“用作短期参考”或者“用作长期参考”，但仅仅是这三种中的一种。当一个参考图像被标识为“用作参考”，那么该参考图像可以被标识为“用作短期参考” 或者 “用作长期参考”，但是不能同时被标识成这两个。

The syntax element adaptive_ref_pic_marking_mode_flag and the content of the decoded reference picture marking syntax structure shall be identical for all coded slices of a coded picture.

语法元素 adaptive_ref_pic_marking_mode_flag 和解码参考图像标识语法结构的内容对于一编码图像的所有编码 slice 来说都是相同的。

The syntax category of the decoded reference picture marking syntax structure shall be inferred as follows.

解码参考图像标识语法结构的语法类别参考如下：

- If the decoded reference picture marking syntax structure is in a slice header, the syntax category of the decoded reference picture marking syntax structure shall be inferred to be equal to 2.
- Otherwise (the decoded reference picture marking syntax structure is in a decoded reference picture marking repetition SEI message as specified in Annex D.), the syntax category of the decoded reference picture marking syntax structure shall be inferred to be equal to 5.
- 如果解码参考图像标识语法结构在 slice 头中，那么解码参考图像标识语法结构的类别等于 2 。
- 否则（解码参考图像标识语法结构在解码参考图像重复标识的 SEI 消息中，如附录 D），解码参考图像标识语法结构的类别等于 5 。

no_output_of_prior_pics_flag specifies how the previously-decoded pictures in the decoded picture buffer are treated after decoding of an IDR picture. See Annex C. When the IDR picture is the first IDR picture in the bitstream, the value of no_output_of_prior_pics_flag has no effect on the decoding process. When the IDR picture is not the first IDR picture in the bitstream and the value of PicWidthInMbs, FrameHeightInMbs, or max_dec_frame_buffering derived from the active sequence parameter set is different from the value of PicWidthInMbs, FrameHeightInMbs, or max_dec_frame_buffering derived from the sequence parameter set active for the preceding sequence, no_output_of_prior_pics_flag equal to 1 may be inferred by the decoder, regardless of the actual value of no_output_of_prior_pics_flag of the active sequence parameter set.

no_output_of_prior_pics_flag 表示解码 IDR 图像后解码缓冲区中前面解码的图像怎么处理，见附录 C。当解码图像是码流中的第一个 IDR 图像，那么 no_output_of_prior_pics_flag 的值对解码过程没有影响。当解码图像不是码流中的第一个 IDR 图像，并且当前序列参数集中的 PicWidthInMbs, FrameHeightInMbs, 或者 max_dec_frame_buffering 和解码前面序列使用的序列参数集中的 PicWidthInMbs, FrameHeightInMbs, 或者 max_dec_frame_buffering 不相同时，那么不管当前序列参数集中的 no_output_of_prior_pics_flag 的值是什么，解码器都将 no_output_of_prior_pics_flag 设置成 1。

long_term_reference_flag equal to 0 specifies that the MaxLongTermFrameIdx variable is set equal to “no long-term frame indices” and that the IDR picture is marked as “used for short-term reference”. long_term_reference_flag equal to 1 specifies that the MaxLongTermFrameIdx variable is set equal to 0 and that the current IDR picture is marked “used for long-term reference” and is assigned LongTermFrameIdx equal to 0.

long_term_reference_flag 等于0 表示变量 MaxLongTermFrameIdx 被设置成“没有长期参考图像”并且 IDR 图像被标识为“用作短期参考”。long_term_reference_flag 等于1 表示变量 MaxLongTermFrameIdx被设置成 0 并且当前图像被标识为“用作长期参考”和 LongTermFrameIdx 被设置成 0。

adaptive_ref_pic_marking_mode_flag selects the reference picture marking mode of the currently decoded picture as specified in Table 7-5.

adaptive_ref_pic_marking_mode_flag 选择当前解码参考图像的标识模式，如 Table 7-5 所示。

Table 7-5 – Interpretation of adaptive_ref_pic_marking_mode_flag

adaptive_ref_pic_marking_mode_flag	Reference picture marking mode specified
0	Sliding window reference picture marking mode: A marking mode providing a first-in first-out mechanism for short-term reference pictures.
1	Adaptive reference picture marking mode: A reference picture marking mode providing syntax elements to specify marking of reference pictures as "unused for reference" and to assign long-term frame indices.

memory_management_control_operation specifies a control operation to be applied to manage the reference picture marking. The memory_management_control_operation syntax element is followed by data necessary for the operation specified by the value of memory_management_control_operation. The values and control operations associated with memory_management_control_operation are specified in Table 7-6.

memory_management_control_operation 表示了应用于管理参考图像标识的控制操作。根据 memory_management_control_operation 的值，memory_management_control_operation 后面接着是完成操作所必要的数据。memory_management_control_operation 的值和控制操作如表 Table 7-6 所示。

memory_management_control_operation shall not be equal to 1 in a slice header unless the specified short-term picture is currently marked as "used for reference" and has not been assigned to a long-term frame index and is not assigned to a long-term frame index in the same decoded reference picture marking syntax structure.

在 slice 头中，memory_management_control_operation 不等于 1 除非被说明的短期图像被标识为“用作参考”并且没有长期帧索引，在相同的解码参考图像标识语法结构中也没有被分配给长期帧索引。

memory_management_control_operation shall not be equal to 2 in a slice header unless the specified long-term picture number refers to a frame or field that is currently marked as "used for reference".

在 slice 头中，memory_management_control_operation 不等于 2 除非被说明的长期图像数表示的是当前被标识为“用作参考”的帧或场。

memory_management_control_operation shall not be equal to 3 in a slice header unless the specified short-term reference picture is currently marked as "used for reference" and has not previously been assigned a long-term frame index and is not assigned to any other long-term frame index within the same decoded reference picture marking syntax structure.

在 slice 头中，memory_management_control_operation 不等于 3 除非被说明的短期参考图像当前被表示为“用作参考”并且以前没有被分配给长期帧索引，在相同的解码参考图像标识语法结构中也没有被分配给任何其他的长期帧索引。

Not more than one memory_management_control_operation equal to 4 shall be present in a slice header.

在 slice 头中不会出现多个等于 4 的 memory_management_control_operation。

memory_management_control_operation shall not be equal to 5 in a slice header unless no memory_management_control_operation in the range of 1 to 3 is present in the same decoded reference picture marking syntax structure.

在 slice 头中，memory_management_control_operation 不等于 3 除非在相同的解码参考图像标识语法结构中没有出现等于 1 到 3 的 memory_management_control_operation。

No more than one memory_management_control_operation shall be present in a slice header that specifies the same action to be taken.

在 slice 中不能出现多个相同操作的 memory_management_control_operation 。

Table 7-6 – Memory management control operation (memory_management_control_operation) values

memory_management_control_operation	Memory Management Control Operation
0	End memory_management_control_operation loop
1	Mark a short-term picture as "unused for reference"
2	Mark a frame or field having a long-term picture number as "unused for reference"
3	Assign a long-term frame index to a short-term picture
4	Specify the maximum long-term frame index
5	Mark all reference pictures as "unused for reference" and set the MaxLongTermFrameIdx variable to "no long-term frame indices"
6	Assign a long-term frame index to the current decoded picture

When decoding a field and a memory_management_control_operation command equal to 3 assigns a long-term frame index to a field that is part of a short-term reference frame or a short-term complementary reference field pair, another memory_management_control_operation command to assign the same long-term frame index to the other field of the same frame or complementary reference field pair shall be present in the same decoded reference picture marking syntax structure.

当解码一场并且等于 3 的 memory_management_control_operation 分配一个长期帧索引给一个短期参考帧的场或者一短期互补参考场对的场时，那么在相同的解码参考图像语法结构中需要另一个 memory_management_control_operation 分配相同的长期索引给一个短期参考帧的另一场或者一短期互补参考场对的另一场。

When the first field (in decoding order) of a complementary reference field pair includes a long_term_reference_flag equal to 1 or a memory_management_control_operation command equal to 6, the decoded reference picture marking syntax structure for the other field of the complementary reference field pair shall contain a memory_management_control_operation command equal to 6 that assigns the same long-term frame index to the other field.

当一互补参考场对的第一场（解码顺序上的第一场）包括一个等于 1 的 long_term_reference_flag 或者等于 6 的 long_term_reference_flag 时，那么对于互补参考场对的另一场来说，解码参考图像标识语法结构中也应该包含一个等于 6 的 memory_management_control_operation，并且分配相同的长期帧索引给另外一场。

difference_of_pic_nums_minus1 is used (with memory_management_control_operation equal to 3 or 1) to assign a long-term frame index to a short-term reference picture or to mark a short-term reference picture as "unused for reference". The resulting picture number derived from difference_of_pic_nums_minus1 shall be a picture number assigned to one of the reference pictures marked as "used for reference" and not previously assigned to a long-term frame index.

difference_of_pic_nums_minus1 被用来（memory_management_control_operation 等于 3 或者 1）分配一长期帧索引给一短期参考图像或者标识一短期参考图像为"不用作参考"。根据 difference_of_pic_nums_minus1 得到的最终图像数（picture number）是一个分配给参考图像的图像数，并且该图像数所表示的图像被标识为"用作参考"，以前也没有被分配给长期帧索引。

The meaning of the resulting picture number is specified as follows.

最终图像数的意义说明如下：

- If field_pic_flag is equal to 0, the resulting picture number shall be one of the set of picture numbers assigned to reference frames or complementary reference field pairs.
- Otherwise (field_pic_flag is equal to 1), the resulting picture number shall be one of the set of picture numbers assigned to reference fields.

- 如果 field_pic_flag 等于 0，那么最终图像数是分配给参考帧或者互补参考场对的图像数。
- 否则（field_pic_flag 等于 1），最终的图像数是分配给参考场的图像数。

long_term_pic_num is used (with memory_management_control_operation equal to 2) to mark a long-term reference picture as "unused for reference". The resulting long-term picture number derived from long_term_pic_num shall be equal to a long-term picture number assigned to one of the reference pictures marked as "used for long-term reference".

long_term_pic_num 被用来（memory_management_control_operation 等于 2）标识一长期参考图像“不用作参考”。根据 long_term_pic_num 得到的最终长期图像数应该等于分配给标识为“用作长期参考”的长期图像数。

The meaning of the resulting long-term picture number is specified as follows.

最终长期图像数的意义说明如下：

- If field_pic_flag is equal to 0, the resulting long-term picture number shall be one of the set of long-term picture numbers assigned to reference frames or complementary reference field pairs.
- Otherwise (field_pic_flag is equal to 1), the resulting long-term picture number shall be one of the set of long-term picture numbers assigned to reference fields.
- 如果 field_pic_flag 等于 0，那么最终长期图像数是分配给参考帧或者互补参考场对的长期图像数。
- 否则（field_pic_flag 等于 1），最终长期图像数是分配给参考场的长期图像数。

long_term_frame_idx is used (with memory_management_control_operation equal to 3 or 6) to assign a long-term frame index to a picture.

long_term_frame_idx 被用来（memory_management_control_operation 等于 3 或者 6）给一图像分配一个长期帧索引。

The presence and value of long_term_frame_idx is constrained as follows.

long_term_frame_idx 值的出现和限制如下：

- If the variable MaxLongTermFrameIdx is equal to “no long-term frame indices”, long_term_frame_idx shall not be present.
- Otherwise (the variable MaxLongTermFrameIdx is not equal to “no long-term frame indices”), the value of long_term_frame_idx shall be in the range of 0 to MaxLongTermFrameIdx, inclusive.
- 如果变量 MaxLongTermFrameIdx 等于 “没有长期帧索引”，那么就没有 long_term_frame_idx；
- 否则（变量 MaxLongTermFrameIdx 不等于 “没有长期帧索引”），那么 long_term_frame_idx 的值应该在 0 到 MaxLongTermFrameIdx 之间，包括这两个数。

max_long_term_frame_idx_plus1 minus 1 specifies the maximum value of long-term frame index allowed for long-term reference pictures (until receipt of another value of max_long_term_frame_idx_plus1). The value of max_long_term_frame_idx_plus1 shall be in the range of 0 to num_ref_frames, inclusive.

max_long_term_frame_idx_plus1 减 1 表示对于长期参考图像的长期帧索引的最大值（知道接收到新的 max_long_term_frame_idx_plus1）。max_long_term_frame_idx_plus1 的范围是 0 到 num_ref_frames，包括这两个数。

7.4.4 Slice data semantics

7.4.4 slice 数据语义

cabac_alignment_one_bit is a bit equal to 1.

cabac_alignment_one_bit 是一个等于 1 的比特位

mb_skip_run specifies the number of consecutive skipped macroblocks for which, when decoding a P or SP slice, mb_type shall be inferred to be P_Skip and the macroblock type is collectively referred to as a P macroblock type, or for which, when decoding a B slice, mb_type shall be inferred to be B_Skip and the macroblock type is collectively referred to as a B macroblock type. The value of mb_skip_run shall be in the range of 0 to PicSizeInMbs – CurrMbAddr, inclusive.

mb_skip_run 表示在解码 P 或者 SP slice 时连续跳过的宏块数目，mb_type 等于 P_Skip，宏块的类型被称为 P 宏块类型，或者表示在解码 B slice 时连续跳过的宏块数目，mb_type 等于 B_Skip，宏块的类型被称为 B 宏块类型，mb_skip_run 值的范围是 0 到 PicSizeInMbs – CurrMbAddr，包括这两个数。

mb_skip_flag equal to 1 specifies that for the current macroblock, when decoding a P or SP slice, mb_type shall be inferred to be P_Skip and the macroblock type is collectively referred to as P macroblock type, or for which, when decoding a B slice, mb_type shall be inferred to be B_Skip and the macroblock type is collectively referred to as B macroblock type. mb_skip_flag equal to 0 specifies that the current macroblock is not skipped.

mb_skip_flag 等于 1 表示在解码 P 或者 SP slice 时当前宏块的 mb_type 应该等于 P_Skip 并且宏块的类型称为 P 宏块，或者表示在解码 B slice 时当前宏块的 mb_type 应该等于 B_Skip 并且宏块的类型称为 B 宏块。mb_skip_flag 等于 0 表示当前宏块没有跳过。

mb_field_decoding_flag equal to 0 specifies that the current macroblock pair is a frame macroblock pair. mb_field_decoding_flag equal to 1 specifies that the macroblock pair is a field macroblock pair. Both macroblocks of a frame macroblock pair are referred to in the text as frame macroblocks, whereas both macroblocks of a field macroblock pair are referred to in the text as field macroblocks.

mb_field_decoding_flag 等于 0 表示当前的宏块对是一个帧宏块对。mb_field_decoding_flag 等于 1 说明当前的宏块对是一场宏块对。帧宏块对中的两个宏块在下面被称为帧宏块，而场宏块对中的两个宏块在下面被称为场宏块。

When mb_field_decoding_flag is not present for either macroblock of a macroblock pair, the value of mb_field_decoding_flag is derived as follows.

对于宏块对中的任意一个宏块来说，当没有 mb_field_decoding_flag 语法元素时，那么 mb_field_decoding_flag 的值计算如下：

– If there is a neighbouring macroblock pair immediately to the left of the current macroblock pair in the same slice, the value of mb_field_decoding_flag shall be inferred to be equal to the value of mb_field_decoding_flag for the neighbouring macroblock pair immediately to the left of the current macroblock pair,
– If there is no neighbouring macroblock pair immediately to the left of the current macroblock pair in the same slice and there is a neighbouring macroblock pair immediately above the current macroblock pair in the same slice, the value of mb_field_decoding_flag shall be inferred to be equal to the value of mb_field_decoding_flag for the neighbouring macroblock pair immediately above the current macroblock pair,
– Otherwise (there is no neighbouring macroblock pair either immediately to the left or immediately above the current macroblock pair in the same slice), the value of mb_field_decoding_flag shall be inferred to be equal to 0.
– 如果在相同的 slice 中当前宏块对的左边有一个相邻的宏块对，那么 mb_field_decoding_flag 应该等于当前宏块对左边相邻宏块对的 mb_field_decoding_flag。
– 如果在相同的 slice 中当前宏块对的左边没有有一个相邻的宏块对，但是在相同的 slice 中当前宏块对的上边有一个相邻的宏块对，那么 mb_field_decoding_flag 应该等于当前宏块对上边相邻宏块对的 mb_field_decoding_flag。
– 否则（在相同的 slice 中当前宏块对的左边和上边都没有有相邻的宏块对），mb_field_decoding_flag 的值将被设置成 0。

end_of_slice_flag equal to 0 specifies that another macroblock is following in the slice. end_of_slice_flag equal to 1 specifies the end of the slice and that no further macroblock follows.

end_of_slice_flag 等于 0 表示在 slice 中接下来是另外一个宏块。end_of_slice_flag 等于 1 表示 slice 的结束，解下来没有本 slice 的宏块。

The function NextMbAddress( ) used in the slice data syntax table is specified in subclause 8.2.2.

在 slice 数据语法表中使用的函数 NextMbAddress( ) 的说明如 8.2.2 小节所示。

7.4.5 Macroblock layer semantics

7.4.5 宏块层的语义

mb_type specifies the macroblock type. The semantics of mb_type depend on the slice type.

mb_type 表示了宏块的类型，mb_type 的语义和 slice 的类型相关。

Tables and semantics are specified for the various macroblock types for I, SI, P, SP, and B slices. Each table presents the value of mb_type, the name of mb_type, the number of macroblock partitions used (given by the NumMbPart(mb_type) function), the prediction mode of the macroblock (if it is not partitioned) or the first partition (given by the MbPartPredMode(mb_type, 0) function) and the prediction mode of the second partition (given by the MbPartPredMode(mb_type, 1) function). When a value is not applicable it is designated by "na". In the text, the value of mb_type may be referred to as the macroblock type and a value X of MbPartPredMode( ) may be referred to in the text by "X macroblock (partition) prediction mode" or as “X prediction macroblocks”.

表和语义表示了在 I, SI, P, SP, 和 B slices 中不同类型的宏块。每个表中包含了 mb_type 的值、mb_type 的名称、宏块部分的数目（由函数 NumMbPart(mb_type) 给出），宏块的预测模式（如果宏块没有被进一步的划分）或者宏块第一部分的预测模式（由函数 MbPartPredMode(mb_type, 0) 给出 ）和宏块第二部分的预测模式（由函数 MbPartPredMode(mb_type, 1) 给出）。表中标为“na”的值表示不可用。在文中，mb_type 的值称为宏块的类型，由 MbPartPredMode( ) 计算出的 X 在文中被称为“X 宏块（部分）预测模式”或者“X 预测宏块”。

Table 7-7 shows the allowed collective macroblock types for each slice_type.

Table 7-7 表示了对于每个 slice_type 类型中的可能宏块类型。

NOTE – There are some macroblock types with Pred_L0 prediction mode that are classified as B macroblock types.

注 － 有一些和 Pred_L0 预测模式相关的宏块类型被划分为 B 宏块类型

Table 7-7 – Allowed collective macroblock types for slice_type

slice_type	allowed collective macroblock types
I (slice)	I (see Table 7-8) (macroblock types)
P (slice)	P (see Table 7-10) and I (see Table 7-8) (macroblock types)
B (slice)	B (see Table 7-11) and I (see Table 7-8) (macroblock types)
SI (slice)	SI (see Table 7-9) and I (see Table 7-8) (macroblock types)
SP (slice)	P (see Table 7-10) and I (see Table 7-8) (macroblock types)

Macroblock types that may be collectively referred to as I macroblock types are specified in Table 7-8.

The macroblock types for I slices are all I macroblock types.

被称为 I 宏块类型的宏块类型的说明在表 Table 7-8 之中。

在 I slice 中宏块的类型都为 I 宏块类型。

Table 7-8 – Macroblock types for I slices

mb_type	Name of mb_type	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	I_4x4	Intra_4x4	na	na	na
1	I_16x16_0_0_0	Intra_16x16	0	0	0
2	I_16x16_1_0_0	Intra_16x16	1	0	0
3	I_16x16_2_0_0	Intra_16x16	2	0	0
4	I_16x16_3_0_0	Intra_16x16	3	0	0
5	I_16x16_0_1_0	Intra_16x16	0	1	0
6	I_16x16_1_1_0	Intra_16x16	1	1	0
7	I_16x16_2_1_0	Intra_16x16	2	1	0
8	I_16x16_3_1_0	Intra_16x16	3	1	0
9	I_16x16_0_2_0	Intra_16x16	0	2	0
10	I_16x16_1_2_0	Intra_16x16	1	2	0
11	I_16x16_2_2_0	Intra_16x16	2	2	0
12	I_16x16_3_2_0	Intra_16x16	3	2	0
13	I_16x16_0_0_1	Intra_16x16	0	0	15
14	I_16x16_1_0_1	Intra_16x16	1	0	15
15	I_16x16_2_0_1	Intra_16x16	2	0	15
16	I_16x16_3_0_1	Intra_16x16	3	0	15
17	I_16x16_0_1_1	Intra_16x16	0	1	15
18	I_16x16_1_1_1	Intra_16x16	1	1	15
19	I_16x16_2_1_1	Intra_16x16	2	1	15
20	I_16x16_3_1_1	Intra_16x16	3	1	15
21	I_16x16_0_2_1	Intra_16x16	0	2	15
22	I_16x16_1_2_1	Intra_16x16	1	2	15
23	I_16x16_2_2_1	Intra_16x16	2	2	15
24	I_16x16_3_2_1	Intra_16x16	3	2	15
25	I_PCM	na	na	na	na

The following semantics are assigned to the macroblock types in Table 7-8:

下面的语义被分配给表 Table 7-8 中的宏块类型。

I_4x4: the macroblock is coded as an Intra_4x4 prediction macroblock.

I_4x4 ：表示宏块采用 Intra_4x4 的预测模式编码。

I_16x16_0_0_0, I_16x16_1_0_0, I_16x16_2_0_0, I_16x16_3_0_0, I_16x16_0_1_0, I_16x16_1_1_0, I_16x16_2_1_0, I_16x16_3_1_0, I_16x16_0_2_0, I_16x16_1_2_0, I_16x16_2_2_0, I_16x16_3_2_0, I_16x16_0_0_1, I_16x16_1_0_1, I_16x16_2_0_1, I_16x16_3_0_1, I_16x16_0_1_1, I_16x16_1_1_1, I_16x16_2_1_1, I_16x16_3_1_1, I_16x16_0_2_1, I_16x16_1_2_1, I_16x16_2_2_1, I_16x16_3_2_1: the macroblock is coded as an Intra_16x16 prediction mode macroblock.

I_16x16_0_0_0, I_16x16_1_0_0, I_16x16_2_0_0, I_16x16_3_0_0, I_16x16_0_1_0, I_16x16_1_1_0, I_16x16_2_1_0, I_16x16_3_1_0, I_16x16_0_2_0, I_16x16_1_2_0, I_16x16_2_2_0, I_16x16_3_2_0, I_16x16_0_0_1, I_16x16_1_0_1, I_16x16_2_0_1, I_16x16_3_0_1, I_16x16_0_1_1, I_16x16_1_1_1, I_16x16_2_1_1, I_16x16_3_1_1, I_16x16_0_2_1, I_16x16_1_2_1, I_16x16_2_2_1, I_16x16_3_2_1：表示宏块采用 Intra_16x16 的预测模式编码。

To each Intra_16x16 prediction macroblock, an Intra16x16PredMode is assigned, which specifies the Intra_16x16 prediction mode. CodedBlockPatternChroma contains the coded block pattern value for chroma as specified in Table 7-12. CodedBlockPatternLuma specifies whether for the luma component non-zero AC transform coefficient levels are present. CodedBlockPatternLuma equal to 0 specifies that there are no AC transform coefficient levels in the luma component of the macroblock. CodedBlockPatternLuma equal to 15 specifies that at least one AC transform coefficient level is in the luma component of the macroblock, requiring scanning of AC transform coefficient levels for all 16 of the 4x4 blocks in the 16x16 block.

对于每一个 Intra_16x16 模式的宏块，分配一个 Intra16x16PredMode，表示使用 Intra_16x16 的预测模式。CodedBlockPatternChroma 表示了色差分量的编码块模式CBP（coded block pattern），如 Table 7-12 小节所示。CodedBlockPatternLuma 表示亮度分量是否有非零的 AC 变换系数。CodedBlockPatternLuma 等于 0 表示在宏块的亮度分量中没有 AC 变换系数。CodedBlockPatternLuma 表示在宏块的亮度分量中至少有一个 AC 变换系数，16X16块中的所有 16 个 4x4 的块都需要扫描 AC 系数。

Intra_4x4 specifies the macroblock prediction mode and specifies that the Intra_4x4 prediction process is invoked as specified in subclause 8.3.1. Intra_4x4 is an Intra macroblock prediction mode.

Intra_4x4 表示了宏块的预测模式，并且使用 8.3.1 小节中的 Intra_4x4 预测过程。Intra_4x4 是帧内宏块预测模式。

Intra_16x16 specifies the macroblock prediction mode and specifies that the Intra_16x16 prediction process is invoked as specified in subclause 8.3.2. Intra_16x16 is an Intra macroblock prediction mode.

Intra_16x16 表示了宏块预测模式，并且使用 8.3.2 小节中的 Intra_16x16 预测过程，Intra_16x16 是帧内宏块预测模式。

For a macroblock coded with mb_type equal to I_PCM, the Intra macroblock prediction mode shall be inferred.

对于 mb_type 等于 I_PCM 的宏块，使用帧内宏块预测模式。

A macroblock type that may be referred to as SI macroblock type is specified in Table 7-9.

被称为 SI 宏块类型的宏块类型的解释见 Table 7-9 。

The macroblock types for SI slices are specified in Table 7-9 and Table 7-8. The mb_type value 0 is specified in Table 7-9 and the mb_type values 1 to 26 are specified in Table 7-8, indexed by subtracting 1 from the value of mb_type.

在SI slice 中宏块类型的说明见 Table 7-9 和 Table 7-8，等于 0 的 mb_type 的说明见 Table 7-9，等于 1 到 26 的 mb_type 的说明见 Table 7-8，用 （mb_type － 1）索引。

Table 7-9 – Macroblock type with value 0 for SI slices

mb_type	Name of mb_type	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	SI	Intra_4x4	na	na	na

The following semantics are assigned to the macroblock type in Table 7-9. The SI macroblock is coded as Intra_4x4 prediction macroblock.

下面的语义被分配给 Table 7-9 中的宏块类型，SI 宏块被编码成 Intra_4x4 预测宏块。

Macroblock types that may be collectively referred to as P macroblock types are specified in Table 7-10.

统称为 P 宏块类型的宏块类型的说明见 Table 7-10。

The macroblock types for P and SP slices are specified in Table 7-10 and Table 7-8. mb_type values 0 to 4 are specified in Table 7-10 and mb_type values 5 to 30 are specified in Table 7-8, indexed by subtracting 5 from the value of mb_type.

在 P 和 SP slice 中宏块类型的说明见 Table 7-10 和 Table 7-8，等于 0 到 4 的 mb_type 的说明见表 Table 7-10，等于 5 到 30 的 mb_type 的说明见 Table 7-8，用 （mb_type － 5）来索引。

Table 7-10 – Macroblock type values 0 to 4 for P and SP slices

mb_type	Name of mb_type	NumMbPart (mb_type)	MbPartPredMode (mb_type, 0)	MbPartPredMode (mb_type, 1)	MbPartWidth (mb_type)	MbPartHeight (mb_type)
0	P_L0_16x16	1	Pred_L0	na	16	16
1	P_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
2	P_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
3	P_8x8	4	na	na	8	8
4	P_8x8ref0	4	na	na	8	8
inferred	P_Skip	1	Pred_L0	na	16	16

The following semantics are assigned to the macroblock types in Table 7-10.

下面的语义被分配给 Table 7-10 中的宏块类型

– P_L0_16x16: the samples of the macroblock are predicted with one luma macroblock partition of size 16x16 luma samples and associated chroma samples.

– P_L0_L0_MxN, with MxN being replaced by 16x8 or 8x16: the samples of the macroblock are predicted using two luma partitions of size MxN equal to 16x8, or two luma partitions of size MxN equal to 8x16, and associated chroma samples, respectively.

– P_8x8: for each sub-macroblock an additional syntax element (sub_mb_type) is present in the bitstream that specifies the type of the corresponding sub-macroblock (see subclause 7.4.5.2).

– P_8x8ref0: has the same semantics as P_8x8 but no syntax element for the reference index (ref_idx_l0) is present in the bitstream and ref_idx_l0[mbPartIdx] shall be inferred to be equal to 0 for all sub-macroblocks of the macroblock (with indices mbPartIdx equal to 0..3).

– P_Skip: no further data is present for the macroblock in the bitstream.

– P_L0_16x16：宏块的象素使用 16x16 大小的亮度象素和相应的色差象素预测得到。

– P_L0_L0_MxN：MxN 可能是 16x8 或者 8x16，宏块的象素使用 16x8 或者 8x16 大小的亮度象素和相应的色差象素预测得到。

– P_8x8：对于每一个子宏块，在码流中有另外的语法元素 sub_mb_type 分别表示对应子宏块的类型（见 7.4.5.2 小节）。

– P_8x8ref0：和 P_8x8 有相同的语义，但是在码流中没有参考索引的语法元素（ref_idx_l0），对于宏块中的所有子宏块来说，它们的 ref_idx_l0[mbPartIdx]（mbPartIdx 等于 0 到 3）都被设置成 0。

– P_Skip：在码流中没有更多关于当前宏块的数据。

The following semantics are assigned to the macroblock prediction modes (MbPartPredMode( )) in Table 7-10.

下面的语义应用于 Table 7-10 中的宏块预测模式(MbPartPredMode( ))。

– Pred_L0: specifies that the inter prediction process is invoked using list 0 prediction. Pred_L0 is an Inter macroblock prediction mode.

– Pred_L0：表示了使用列表 0 预测的帧间预测过程，Pred_L0 是帧间宏块预测模式。

Macroblock types that may be collectively referred to as B macroblock types are specified in Table 7-11.

统称为 B 宏块类型的宏块类型的说明见 Table 7-11。

The macroblock types for B slices are specified in Table 7-11 and Table 7-8. The mb_type values 0 to 22 are specified in Table 7-11 and the mb_type values 23 to 48 are specified in Table 7-8, indexed by subtracting 23 from the value of mb_type.

B slices 中宏块类型的说明见 Table 7-11 and Table 7-8，等于 0 到 22 的 mb_type 的说明见 Table 7-11，等于 23 到 48 的 mb_type 的说明见 Table 7-8，用（mb_type － 23）索引。

Table 7-11 – Macroblock type values 0 to 22 for B slices

mb_type	Name of mb_type	NumMbPart (mb_type)	MbPartPredMode (mb_type, 0)	MbPartPredMode (mb_type, 1)	MbPartWidth (mb_type)	MbPartHeight (mb_type)
0	B_Direct_16x16	na	Direct	na	8	8
1	B_L0_16x16	1	Pred_L0	na	16	16
2	B_L1_16x16	1	Pred_L1	na	16	16
3	B_Bi_16x16	1	BiPred	na	16	16
4	B_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
5	B_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
6	B_L1_L1_16x8	2	Pred_L1	Pred_L1	16	8
7	B_L1_L1_8x16	2	Pred_L1	Pred_L1	8	16
8	B_L0_L1_16x8	2	Pred_L0	Pred_L1	16	8
9	B_L0_L1_8x16	2	Pred_L0	Pred_L1	8	16
10	B_L1_L0_16x8	2	Pred_L1	Pred_L0	16	8
11	B_L1_L0_8x16	2	Pred_L1	Pred_L0	8	16
12	B_L0_Bi_16x8	2	Pred_L0	BiPred	16	8
13	B_L0_Bi_8x16	2	Pred_L0	BiPred	8	16
14	B_L1_Bi_16x8	2	Pred_L1	BiPred	16	8
15	B_L1_Bi_8x16	2	Pred_L1	BiPred	8	16
16	B_Bi_L0_16x8	2	BiPred	Pred_L0	16	8
17	B_Bi_L0_8x16	2	BiPred	Pred_L0	8	16
18	B_Bi_L1_16x8	2	BiPred	Pred_L1	16	8
19	B_Bi_L1_8x16	2	BiPred	Pred_L1	8	16
20	B_Bi_Bi_16x8	2	BiPred	BiPred	16	8
21	B_Bi_Bi_8x16	2	BiPred	BiPred	8	16
22	B_8x8	4	na	na	8	8
inferred	B_Skip	na	Direct	na	8	8

The following semantics are assigned to the macroblock types in Table 7-11:

下面的语义分配给 Table 7-11 中的宏块类型：

– B_Direct_16x16: no motion vector differences or reference indices are present for the macroblock in the bitstream. The functions MbPartWidth(B_Direct_16x16), and MbPartHeight(B_Direct_16x16) are used in the derivation process for motion vectors and reference frame indices in subclause 8.4.1 for direct mode prediction.

– B_Direct_16x16：在码流中没有关于宏块的运动矢量残差或者参考索引。对于直接模式预测，使用 8.4.1 小节中的过程来计算运动矢量和参考索引。函数 MbPartWidth(B_Direct_16x16), 和 MbPartHeight(B_Direct_16x16) 在 8.4.1 小节中使用。

– B_X_16x16 with X being replaced by L0, L1, or Bi: the samples of the macroblock are predicted with one luma macroblock partition of size 16x16 luma samples and associated chroma samples. For a macroblock with type B_X_16x16 with X being replaced by either L0 or L1, one motion vector difference and one reference index is present in the bitstream for the macroblock. For a macroblock with type B_X_16x16 with X being replaced by Bi, two motion vector differences and two reference indices are present in the bitstream for the macroblock.

– B_X_16x16：X 可能是 L0, L1, 或者 Bi，宏块的象素使用 16x16 的亮度象素和对应的色差象素预测得到。对于类型为 B_X_16x16 的宏块，X 可能是 L0 或者 L1，在码流有当前宏块的运动矢量残差和参考索引的语法元素。对于 类型为 B_X_16x16 的宏块，X 是 Bi，在码流有当前宏块的两个运动矢量残差和两个参考索引的语法元素。

– B_X0_X1_MxN, with X0, X1 referring to the first and second macroblock partition and being replaced by L0, L1, or Bi, and MxN being replaced by 16x8 or 8x16: the samples of the macroblock are predicted using two luma partitions of size MxN equal to 16x8, or two luma partitions of size MxN equal to 8x16, and associated chroma samples, respectively. For a macroblock partition X0 or X1 with X0 or X1 being replaced by either L0 or L1, one motion vector difference and one reference index is present in the bitstream. For a macroblock partition X0 or X1 with X0 or X1 being replaced by Bi, two motion vector differences and two reference indices are present in the bitstream for the macroblock partition.

– B_X0_X1_MxN：X0, X1表示宏块的第一部分和第二部分，并且可以被 L0, L1, 或者 Bi 取代，MxN 可以被 16x8 或者 8x16 取代，宏块的亮度象素用两个大小为 16x8 的亮度象素块或者 两个大小为 8x16 的亮度象素块和对应的色差块预测得到。对于宏块部分 X0 或者 X1（X0 或 X1 可以被 L0 或 L1 取代），在码流中将有一个对应的运动矢量和一个对应的参考索引。对于宏块部分 X0 或者 X1（X0 或 X1 可以被 Bi 取代），在码流中将有两个对应的运动矢量和两个个对应的参考索引。

– B_8x8: for each sub-macroblock an additional syntax element (sub_mb_type) is present in the bitstream that specifies the type of the corresponding sub-macroblock (see subclause 7.4.5.2).

– B_8x8：对于每个子宏块，码流中将有另外的语法元素(sub_mb_type)来说明对应子宏块的类型。

– B_Skip: no further data is present for the macroblock in the bitstream. The functions MbPartWidth(B_Skip), and MbPartHeight(B_Skip) are used in the derivation process for motion vectors and reference frame indices in subclause 8.4.1 for direct mode prediction.

– B_Skip：在码流中没有更多的关于当前宏块的数据。对于直接模式预测，使用 8.4.1 小节中的过程来计算运动矢量和参考索引。函数 MbPartWidth(B_Skip) 和 MbPartHeight(B_Skip) 在 8.4.1 小节中使用。

The following semantics are assigned to the macroblock prediction modes (MbPartPredMode( )) in Table 7-11.

下面的语义被分配给 Table 7-11 中的宏块预测模式。

– Direct: motion vector differences or reference indices are present for the macroblock (in case of B_Skip or B_Direct_16x16) in the bitstream. Direct is an Inter macroblock prediction mode.

– Pred_L0: see semantics for Table 7-10.

– Pred_L1: specifies that the Inter prediction process is invoked using list 1 prediction. Pred_L1 is an Inter macroblock prediction mode.

– BiPred: specifies that the Inter prediction process is invoked using list 0 and list 1 prediction. BiPred is an Inter macroblock prediction mode.

– Direct：在码流中包含有当前宏块的运动矢量残差或者参考索引（在 B_Skip 或 B_Direct_16x16 情况下）。Direct 是一种帧间宏块预测模式。

– Pred_L0：见表 Table 7-10 中的语义。

– Pred_L1：表示了利用列表 1 的帧间预测的过程，Pred_L1 是宏块帧间预测模式。

– BiPred：表示了利用列表 0 和列表 1 的帧间预测过程，BiPred 是宏块帧间预测模式。

pcm_alignment_zero_bit is a bit equal to 0.

pcm_alignment_zero_bit 为等于 0 的比特位。

pcm_byte[i] is a sample value. pcm_byte[i] shall not be equal to 0. The first 256 pcm_byte[i] values represent luma sample values in the raster scan within the macroblock. The next (256 * (ChromaFormatFactor - 1)) / 2 pcm_byte[i] values represent Cb sample values in the raster scan within the macroblock. The last (256 * (ChromaFormatFactor - 1)) / 2 pcm_byte[i] values represent Cr sample values in the raster scan within the macroblock.

pcm_byte[i] 是一个象素值。pcm_byte[i] 不等于 0，最初的 256 个 pcm_byte[i] 值表示了宏块内按照栅格扫描顺序的亮度象素值，接下来的 (256 * (ChromaFormatFactor - 1)) / 2 pcm_byte[i] 表示了宏块内按照栅格扫描顺序的色差 Cb 的象素值，最后的 (256 * (ChromaFormatFactor - 1)) / 2 pcm_byte[i] 表示了宏块内按照栅格扫描顺序的色差 Cr 的象素值。

coded_block_pattern specifies which of the six 8x8 blocks - luma and chroma - contain non-zero transform coefficient levels. For macroblocks with prediction mode not equal to Intra_16x16, coded_block_pattern is present in the bitstream and the variables CodedBlockPatternLuma and CodedBlockPatternChroma are derived as follows.

coded_block_pattern 表示了 6 个 8x8 块中（色差和亮度）哪个块有非零的变换系数。对于预测模式等于 Intra_16x16 的宏块，coded_block_pattern 出现在码流中，并且变量 CodedBlockPatternLuma 和 CodedBlockPatternChroma 的计算如下：

CodedBlockPatternLuma = coded_block_pattern % 16
CodedBlockPatternChroma = coded_block_pattern / 16 (7-22)

The meaning of CodedBlockPatternChroma is given in Table 7-12.

CodedBlockPatternChroma 的意义见 Table 7-12

Table 7-12 – Specification of CodedBlockPatternChroma values

CodedBlockPatternChroma	Description
0	All chroma transform coefficient levels are equal to 0.
1	One or more chroma DC transform coefficient levels are non-zero. All chroma AC transform coefficient levels are equal to 0.
2	Zero or more chroma DC transform coefficient levels are non-zero valued. One or more chroma AC transform coefficient levels are non-zero valued.

mb_qp_delta can change the value of QP_Y in the macroblock layer. The decoded value of mb_qp_delta shall be in the range of -26 to +25, inclusive. mb_qp_delta shall be inferred to be equal to 0 if it is not present for any macroblock (including P_Skip and B_Skip macroblock types).

mb_qp_delta 可以改变宏块层的 QP_Y 的值。mb_qp_delta 的解码值应该在 -26 到 +25 之间，包括 -26 和 +25 者两个数。如果在宏块层的码流中没有 mb_qp_delta 语法元素，那么 mb_qp_delta 的值被设置成 0。

The value of QP_Y is derived as

QP_Y 的值计算如下：

$$QP_Y = (QP_{Y,PREV} + mb\_qp\_delta + 52) \% 52 \quad (7\text{-}23)$$

where $QP_{Y,PREV}$ is the luma quantisation parameter, QP_Y, of the previous macroblock in the current slice. For the first macroblock in the slice $QP_{Y,PREV}$ is initially set equal to $SliceQP_Y$ derived in Equation 7-16 at the start of each slice.

这里 $QP_{Y,PREV}$ 是当前 slice 中前一宏块的量化参数 QP_Y 的值，在每个 slice 的开始处，对于 slice 中的第一个宏块，$QP_{Y,PREV}$ 应该被初始化成等式 7-16 中的 $SliceQP_Y$ 值。

7.4.5.1 Macroblock prediction semantics

7.4.5.1 宏块预测的语义

All samples of the macroblock are predicted. The prediction modes are derived using the following syntax elements.

宏块中所有的象素都是预测得到的，预测模式的计算使用下面的语法元素。

prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]** and **rem_intra4x4_pred_mode[** luma4x4BlkIdx **]** specify the Intra_4x4 prediction of the 4x4 luma block with index luma4x4BlkIdx = 0..15.

prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]** 和 **rem_intra4x4_pred_mode[** luma4x4BlkIdx **]** 表示了 4x4 亮度块的 Intra_4x4 预测，其中的索引 luma4x4BlkIdx = 0..15。

intra_chroma_pred_mode specifies the type of spatial prediction used for chroma whenever any part of the luma macroblock is intra coded, as shown in Table 7-13.

intra_chroma_pred_mode 表示了色差的空间预测类型，只要亮度宏块的任何一部分使用了帧内编码，如 Table 7-13 所示。

Table 7-13 – Relationship between intra_chroma_pred_mode and spatial prediction modes

intra_chroma_pred_mode	Intra Chroma Prediction Mode
0	DC
1	Horizontal
2	Vertical
3	Plane

ref_idx_l0[mbPartIdx **]** when present, specifies the index in list 0 of the reference picture to be used for prediction.

ref_idx_l0[mbPartIdx **]** 如果出现的话，表示了用作预测的列表 0 中的参考图像的索引值。

The range of ref_idx_l0[mbPartIdx], the index in list 0 of the reference picture, and, if applicable, the parity of the field within the reference picture used for prediction are specified as follows.

ref_idx_l0[mbPartIdx] 的范围，参考图像列表 0 中索引和用作参考的参考图像中对应场的说明：

- If MbaffFrameFlag is equal to 0 or mb_field_decoding_flag is equal to 0, the value of ref_idx_l0[mbPartIdx] shall be in the range of 0 to num_ref_idx_l0_active_minus1, inclusive.
- Otherwise (MbaffFrameFlag is equal to 1 and mb_field_decoding_flag is equal to 1), the value of ref_idx_l0[mbPartIdx] shall be in the range of 0 to 2 * num_ref_idx_l0_active_minus1 + 1, inclusive.
- 如果 MbaffFrameFlag 等于 0 或者 mb_field_decoding_flag 等于 0，那么 ref_idx_l0[mbPartIdx] 的范围是 0 到 num_ref_idx_l0_active_minus1，包括这两个数。
- 否则（MbaffFrameFlag 等于 1 并且 mb_field_decoding_flag 等于 1），那么 ref_idx_l0[mbPartIdx] 的范围是 0 到 2 * num_ref_idx_l0_active_minus1 + 1，包括这两个数。

When only one reference picture is used for inter prediction, the values of ref_idx_l0[mbPartIdx] shall be inferred to be equal to 0.

当帧间预测仅仅使用一个参考图像时，那么 ref_idx_l0[mbPartIdx] 的值应该被设置成 0 。

ref_idx_l1[mbPartIdx **]** has the same semantics as ref_idx_l0, with l0 and list 0 replaced by l1 and list 1, respectively.

ref_idx_l1[mbPartIdx **]** 和 ref_idx_l0 有相同的语义，l1 和列表 1 分别替代 l0 和列表 0。

mvd_l0[mbPartIdx **][** 0 **][** compIdx **]** specifies the difference between a vector component to be used and its prediction. The index mbPartIdx specifies to which macroblock partition mvd_l0 is assigned. The partitioning of the macroblock is specified by mb_type. The horizontal motion vector component difference is decoded first in decoding order and is assigned CompIdx = 0. The vertical motion vector component is decoded second in decoding order and is assigned CompIdx = 1. The range of the components of mvd_l0[mbPartIdx][0][compIdx] is specified by constraints on the motion vector variable values derived from it as specified in Annex A.

mvd_l0[mbPartIdx **][** 0 **][** compIdx **]** 表示了实际使用的运动矢量和它的预测值之间的差。MbPartIdx 表示了 mvd_l0 属于宏块的哪个部分，macroblock 部分由 mb_type 表示，水平运动矢量分量差先被解码并且 CompIdx = 0，然后解码垂直运动矢量分量并且 CompIdx = 1。附录 A 中对运动矢量变量值的限制说明了 mvd_l0[mbPartIdx][0][compIdx] 分量的范围。

mvd_l1[mbPartIdx **][** 0 **][** compIdx **]** has the same semantics as mvd_l0, with l0 and L0 replaced by l1 and L1, respectively.

mvd_l1[mbPartIdx **][** 0 **][** compIdx **]** 和 mvd_l0 有相同的语义，用 l1 和 L1 分别替代 l0 和 L0。

7.4.5.2 Sub-macroblock prediction semantics

7.4.5.2 子宏块预测的语义

sub_mb_type[mbPartIdx **]** specifies the sub-macroblock types.

sub_mb_type[mbPartIdx **]** 表示了子宏块的类型。

Tables and semantics are specified for the various sub-macroblock types for P, SP, and B slices. Each table presents the value of sub_mb_type, the name of sub_mb_type, the number of sub-macroblock partitions used (given by the NumSubMbPart(sub_mb_type) function), and the prediction mode of the sub-macroblock (given by the SubMbPredMode(sub_mb_type) function). In the text, the value of sub_mb_type may be referred to by "sub-macroblock type". In the text, the value of SubMbPredMode( ) may be referred to by "sub-macroblock prediction mode".

用表和语义表示了 P, SP, and B slices 中不同的子宏块类型。每一个表表示了 sub_mb_type 的值，sub_mb_type 的名称、子宏块部分的数目（由函数 NumSubMbPart(sub_mb_type) 得到），和子宏块的预测模式（由函数 SubMbPredMode(sub_mb_type) 预测得到）。在本文中，sub_mb_type 的值被称为"子宏块类型"，函数 SubMbPredMode( ) 的返回值被称为"子宏块预测模式"。

The sub-macroblock types for P macroblock types are specified in Table 7-14.

对于 P 宏块类型，子宏块的类型说明如 Table 7-14。

Table 7-14 – Sub-macroblock types in P macroblocks

sub_mb_type[mbPartIdx]	Name of sub_mb_type[mbPartIdx]	NumSubMbPart (sub_mb_type[mbPartIdx])	SubMbPredMode (sub_mb_type[mbPartIdx])	SubMbPartWidth (sub_mb_type[mbPartIdx])	SubMbPartHeight (sub_mb_type[mbPartIdx])
0	P_L0_8x8	1	Pred_L0	8	8
1	P_L0_8x4	2	Pred_L0	8	4
2	P_L0_4x8	2	Pred_L0	4	8
3	P_L0_4x4	4	Pred_L0	4	4

The following semantics are assigned to the sub-macroblock types in Table 7-14.

下面的语义用来说明 Table 7-14 中的子宏块类型。

– P_L0_8x8: the samples of the sub-macroblock are predicted with one luma sub-macroblock partition of size 8x8 luma samples and associated chroma samples.

– P_L0_L0_MxN, with MxN being replaced by 8x4, 4x8, or 4x4: the samples of the sub-macroblock are predicted using two luma partitions of size MxN equal to 8x4, or two luma partitions of size MxN equal to 4x8, or four luma partitions of size MxN equal to 4x4, and associated chroma samples, respectively.

– P_L0_8x8：子宏块的象素使用 8x8 的亮度块象素和对应的色差象素预测得到。

– P_L0_L0_MxN：MxN 可能是 8x4, 4x8, 或者 4x4，子宏块的象素使用两个 8x4 的亮度象素，或者两个 4x8 的亮度象素，或者四个 4x4 的亮度象素和对应的色差象素预测得到。

The following semantics are assigned to the sub-macroblock prediction modes (SubMbPredMode( )) in Table 7-14.

下面的语义表示了子宏块的预测模式（SubMbPredMode( )），见表 Table 7-14。

– Direct: specifies that no motion vector differences or reference indices are present for the sub-macroblock (in case of B_Direct_8x8) in the bitstream. Direct is an Inter macroblock prediction mode.

– Pred_L0: see semantics for Table 7-10.

– Pred_L1: see semantics for Table 7-11.

– BiPred: see semantics for Table 7-11.

– Direct：对于子宏块来说，码流中没有运动矢量残差和参考索引的语法元素，（在 B_Direct_8x8 的情况下），Direct 是帧间宏块预测模式。

– Pred_L0：见 Table 7-10 中的语义。

– Pred_L1：见 Table 7-11 中的语义。

– BiPred：见 Table 7-11 中的语义。

The sub-macroblock types for B macroblock types are specified in Table 7-15.

对于 B 宏块类型中的子宏块类型的说明，见 Table 7-15。

Table 7-15 – Sub-macroblock types in B macroblocks

sub_mb_type[mbPartIdx]	Name of sub_mb_type[mbPartIdx]	NumSubMbPart (sub_mb_type[mbPartIdx])	SubMbPredMode (sub_mb_type[mbPartIdx])	SubMbPartWidth (sub_mb_type[mbPartIdx])	SubMbPartHeight (sub_mb_type[mbPartIdx])
0	B_Direct_8x8	na	Direct	4	4
1	B_L0_8x8	1	Pred_L0	8	8
2	B_L1_8x8	1	Pred_L1	8	8
3	B_Bi_8x8	1	BiPred	8	8
4	B_L0_8x4	2	Pred_L0	8	4
5	B_L0_4x8	2	Pred_L0	4	8
6	B_L1_8x4	2	Pred_L1	8	4
7	B_L1_4x8	2	Pred_L1	4	8
8	B_Bi_8x4	2	BiPred	8	4
9	B_Bi_4x8	2	BiPred	4	8
10	B_L0_4x4	4	Pred_L0	4	4
11	B_L1_4x4	4	Pred_L1	4	4
12	B_Bi_4x4	4	BiPred	4	4

The following semantics are assigned to the macroblock types in Table 7-15:

下面的语义说明了 Table 7-15 中的宏块类型：

– B_Direct_8x8: no motion vector differences or reference indices are present for the sub-macroblock in the bitstream. The functions SubMbPartWidth(B_Direct_8x8) and SubMbPartHeight(B_Direct_8x8) are used in the derivation process for motion vectors and reference frame indices in subclause 8.4.1 for direct mode prediction.

– B_Direct_8x8：在码流中没有与此子宏块相对应的运动矢量残差或者参考索引。对于直接模式预测，使用 8.4.1 小节的处理过程来计算出运动矢量和参考帧索引，函数 SubMbPartWidth(B_Direct_8x8) 和 SubMbPartHeight(B_Direct_8x8) 在 8.4.1 小节中使用。

– B_X_MxN, with X being replaced by L0, L1, or Bi, and MxN being replaced by 8x8, 8x4, 4x8 or 4x4: the samples of the sub-macroblock are predicted using one luma partition of size MxN equal to 8x8, or the samples of the sub-macroblock are predicted using two luma partitions of size MxN equal to 8x4, or the samples of the sub-macroblock are predicted using two luma partitions of size MxN equal to 4x8, or the samples of the sub-macroblock are predicted using four luma partitions of size MxN equal to 4x4, and associated chroma samples, respectively. All sub-macroblock partitions share the same reference index. For an MxN sub-macroblock partition in a sub-macroblock with sub_mb_type being B_X_MxN with X being replaced by either L0 or L1, one motion vector difference is present in the bitstream. For an MxN sub-macroblock partition in a sub-macroblock with sub_mb_type being B_Bi_MxN, two motion vector difference are present in the bitstream.

– B_X_MxN：X 可能是 L0, L1, 或者 Bi，MxN 可能是 8x8, 8x4, 4x8 或者 4x4，子宏块的象素使用一个 8x8 的亮度象素块、或者两个 8x4 的亮度象素块、或者两个 4x8 的亮度象素块、或者四个 4x4 的亮度象素块和对应的色差块预测得到。所有的子宏块部分都使用相同的参考索引。对于子宏块中 sub_mb_type 等于 B_X_MxN（X 可能被 L0 或者 L1 替代）的一个 MxN 的子宏块部分来说，码流中有一个运动矢量残差。对于子宏块中 sub_mb_type 等于 B_Bi_MxN 的一个 MxN 的子宏块部分来说，码流中有两个运动矢量残差。

The following semantics are assigned to the sub-macroblock prediction modes (SubMbPredMode( )) in Table 7-15.

下面的语义表示了子宏块的预测模式（SubMbPredMode( )），见 Table 7-15 所示。

– Direct: see semantics for Table 7-11.

– Pred_L0: see semantics for Table 7-10.

– Pred_L1: see semantics for Table 7-11.

– BiPred: see semantics for Table 7-11.

– Direct: 见 Table 7-11 中的语义

– Pred_L0: 见 Table 7-10 中的语义

– Pred_L1: 见 Table 7-11 中的语义

– BiPred: 见 Table 7-11 中的语义

ref_idx_l0[mbPartIdx **]** has the same semantics as ref_idx_l0 in subclause 7.4.5.1.

ref_idx_l0[mbPartIdx **]** 和 7.4.5.1 小节中的 ref_idx_l0 有相同的语义。

ref_idx_l1[mbPartIdx **]** has the same semantics as ref_idx_l1 in subclause 7.4.5.1.

ref_idx_l1[mbPartIdx **]** 和 7.4.5.1 小节中的 ref_idx_l1 有相同的语义。

mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]** has the same semantics as mvd_l0 in subclause 7.4.5.1, except that it is applied to the sub-macroblock partition index with subMbPartIdx. The indices mbPartIdx and subMbPartIdx specify to which macroblock partition and sub-macroblock partition mvd_l0 is assigned.

mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]** 和 7.4.5.1 中的 mvd_l0 有相同的语义，除了它应用于索引为 subMbPartIdx 的子宏块部分。索引 mbPartIdx 和 subMbPartIdx 表示了 mvd_l0 属于哪个宏块部分或者哪个子宏块部分。

mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]** has the same semantics as mvd_l1 in subclause 7.4.5.1.

mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]** 和 7.4.5.1 小节中的 mvd_l1 有相同的语义。

7.4.5.3 Residual data semantics

7.4.5.3 残差数据的语义

The syntax structure residual_block( ), which is used for parsing the transform coefficient levels, is assigned as follows.

用来解析变换系数值的语法结构 residual_block( ) 的说明如下：

- If entropy_coding_mode_flag is equal to 0, residual_block is set equal to residual_block_cavlc, which is used for parsing the syntax elements for transform coefficient levels.
- Otherwise (entropy_coding_mode_flag is equal to 1), residual_block is set equal to residual_block_cabac, which is used for parsing the syntax elements for transform coefficient levels.
- 如果 entropy_coding_mode_flag 等于 0， 那么 residual_block 被设置成 residual_block_cavlc，被用来解析变换系数的语法元素。
- 否则（如果 entropy_coding_mode_flag 等于 1）， residual_block 被设置成 residual_block_cabac，被用来解析变换系数的语法元素。

Depending on mb_type, luma or chroma, the syntax structure residual_block(coeffLevel, maxNumCoeff) is used with the arguments coeffLevel, which is a list containing the maxNumCoeff transform coefficient levels that are parsed in residual_block( ), and maxNumCoeff as follows.

根据 mb_type 的值，亮度或者色差，使用语法结构 residual_block(coeffLevel, maxNumCoeff)，其中参数 coeffLevel 表示在residual_block( ) 中解析出来的 maxNumCoeff 个变换系数值的列表，maxNumCoeff 的计算如下：

- If MbPartPredMode(mb_type, 0) is equal to Intra_16x16, the transform coefficient levels are parsed into the list Intra16x16DCLevel and into the 16 lists Intra16x16ACLevel[i]. Intra16x16DCLevel contains the 16 transform coefficient levels of the DC transform coefficient levels for each 4x4 luma block. For each of the 16 4x4 luma blocks indexed by i = 0..15, the 15 AC transform coefficients levels of the i-th block are parsed into the i-th list Intra16x16ACLevel[i].
- 如果 MbPartPredMode(mb_type, 0) 等于 Intra_16x16，变换系数值被解析到Intra16x16DCLevel 和 16个 Intra16x16ACLevel[i] 列表中，Intra16x16DCLevel 中含有 16 变换系数值，它们分别对应于16个 4x4 亮度块 DC 变换系数值。对于每一个索引为 i 的 4x4 亮度块来说，第 i 个块的 15 个 AC 系数值被解析到第 i 个列表 Intra16x16ACLevel[i] 中。
- Otherwise (MbPartPredMode(mb_type, 0) is not equal to Intra_16x16), for each of the 16 4x4 luma blocks indexed by i = 0..15, the 16 transform coefficient levels of the i-th block are parsed into the i-th list LumaLevel[i].
- 否则（MbPartPredMode(mb_type, 0) 不等于 Intra_16x16），对于每一个索引为 i 的 4x4 亮度块来说，第 i 个块的 16 个变换系数值被解析到第 i 个列表 LumaLevel[i] 中。
- For each chroma component, indexed by iCbCr = 0..1, the 4 DC transform coefficient levels of the 4x4 chroma blocks are parsed into iCbCr-th list ChromaDCLevel[iCbCr].
- 对于每一个色差分量，索引为 iCbCr = 0..1，4x4 色差块的 4 个 DC 变换系数值被解析到第 iCbCr-th 个列表 ChromaDCLevel[iCbCr] 中。
- For each of the 4x4 chroma blocks, indexed by i4x4 = 0..3, of each chroma component, indexed by iCbCr = 0..1, the 15 AC transform coefficient levels are parsed into the i4x4-th list of the iCbCr-th chroma component ChromaACLevel[iCbCr][i4x4].
- 对于每一个 4x4 的色差块，索引 i4x4 = 0..3，色差分量为 iCbCr = 0..1，15 个 AC 变换系数被解析到第 iCbCr 个色差分量的第 i4x4 个列表 ChromaACLevel[iCbCr][i4x4] 中。

7.4.5.3.1 Residual block CAVLC semantics

7.4.5.3.1 残差块 CAVLC 的语义

The function TotalCoeff(coeff_token) that is used in subclause 7.3.5.3.1 returns the number of non-zero transform coefficient levels derived from coeff_token.

根据 coeff_token，本小节使用的函数 TotalCoeff(coeff_token) 返回了非零变换系数的个数。

The function TrailingOnes(coeff_token) that is used in subclause 7.3.5.3.1 returns the trailing ones derived from coeff_token.

根据 coeff_token，本小节使用的函数 TrailingOnes(coeff_token) 返回了 trailing one 个数。

coeff_token specifies the total number of non-zero transform coefficient levels and the number of trailing one transform coefficient levels in a transform coefficient level scan. A trailing one transform coefficient level is one of up to three consecutive non-zero transform coefficient levels having an absolute value equal to 1 at the end of a scan of non-zero transform coefficient levels. The range of coeff_token is specified in subclause 9.2.1.

coeff_token 表示了非零变换系数值的总数和在变换系数值扫描过程中 trailing one 变换系数的个数。一个 trailing one 变换系数是在扫描非零变换系数的结尾处最多三个绝对值等于 1 的连续的非零变换系数中的一个。coeff_token 值的范围由 9.2.1 小节说明。

trailing_ones_sign_flag specifies the sign of a trailing one transform coefficient level as follows.

trailing_ones_sign_flag 表示了 trailing one 变换系数的符号，如下所示：

- If trailing_ones_sign_flag is equal to 0, the corresponding transform coefficient level is decoded as +1.
- Otherwise (trailing_ones_sign_flag equal to 1), the corresponding transform coefficient level is decoded as -1.
- 如果 trailing_ones_sign_flag 等于 0，对应的变换系数值为 +1。
- 否则（trailing_ones_sign_flag 等于 1），对应的变换系数值为 –1。

level_prefix and **level_suffix** specify the value of a non-zero transform coefficient level. The range of level_prefix and level_suffix is specified in subclause 9.2.2.

level_prefix 和 **level_suffix** 表示了非零变换系数的值。level_prefix 和 level_suffix 的范围如 9.2.2 小节所示。

total_zeros specifies the total number of zero-valued transform coefficient levels that are located before the position of the last non-zero transform coefficient level in a scan of transform coefficient levels. The range of total_zeros is specified in subclause 9.2.3.

total_zeros 表示了按照变换系数的扫描顺序在最后一个非零变换系数之前的零值变换系数的总数。total_zeros 的范围如 9.2.3 小节所示。

run_before specifies the number of consecutive transform coefficient levels in the scan with zero value before a non-zero valued transform coefficient level. The range of run_before is specified in subclause 9.2.3.

run_before 表示了在一个非零变换系数之前连续的零值变换系数的个数。run_before 的范围如 9.2.3 小节所示。

coeffLevel contains maxNumCoeff transform coefficient levels for the current list of transform coefficient levels.

coeffLevel 包含了当前变换系数列表中 maxNumCoeff 个变换系数值。

7.4.5.3.2 Residual block CABAC semantics

7.4.5.3.2 残差块 CABAC 的语义

coded_block_flag specifies whether the block contains non-zero transform coefficient levels as follows.

coded_block_flag 表示了当前块是否含有非零变换系数值，如下所示：

- If coded_block_flag is equal to 0, the block contains no non-zero transform coefficient levels.
- Otherwise (coded_block_flag is equal to 1), the block contains at least one non-zero transform coefficient level.
- 如果 coded_block_flag 等于 0，那么当前块不含有非零的变换系数值；
- 否则（coded_block_flag 等于 1），当前块至少含有一个非零的变换系数值；

significant_coeff_flag[i **]** specifies whether the transform coefficient level at scanning position i is non-zero as follows.

significant_coeff_flag[i **]** 表示了扫描位置处的变换系数值是否为非零值，如下所示：

- If significant_coeff_flag[i] is equal to 0, the transform coefficient level at scanning position i is set equal to 0;
- Otherwise (significant_coeff_flag[i] is equal to 1), the transform coefficient level at scanning position i has a non-zero value.
- 如果 significant_coeff_flag[i] 等于 0，那么扫描位置 i 处的变换系数值等于 0；
- 否则（significant_coeff_flag[i] 等于 1），扫描位置 i 处的变换系数值不等于 0；

last_significant_coeff_flag[i **]** specifies for the scanning position i whether there are non-zero transform coefficient levels for subsequent scanning positions i + 1 to maxNumCoeff – 1 as follows.

last_significant_coeff_flag[i **]** 表示了在扫描位置 i + 1 一直到 maxNumCoeff – 1 处是否有非零变换系数值。

- If last_significant_coeff_flag[i] is equal to 1, all following transform coefficient levels (in scanning order) of the block have value equal to 0..
- Otherwise (last_significant_coeff_flag[i] is equal to 0), there are further non-zero transform coefficient levels along the scanning path.
- 如果 last_significant_coeff_flag[i] 等于 1，当前块中接下来的（按照扫描顺序）变换系数值全部为 0；
- 否则（last_significant_coeff_flag[i] 等于 0），沿着扫描顺序还有非零的变换系数值。

coeff_abs_level_minus1[i **]** is the absolute value of a transform coefficient level minus 1. The value of coeff_abs_level_minus1 is constrained by the limits in subclause 8.5.

coeff_abs_level_minus1[i **]** 表示了变换系数值减 1 的绝对值。coeff_abs_level_minus1值的限制见 8.5 小节所示。

coeff_sign_flag[i **]** specifies the sign of a transform coefficient level as follows.

coeff_sign_flag[i **]** 表示了变换系数值的符号，如下所示：

- If coeff_sign_flag is equal to 0, the corresponding transform coefficient level has a positive value.
- Otherwise (coeff_sign_flag is equal to 1), the corresponding transform coefficient level has a negative value.
- 如果 coeff_sign_flag 等于 0，对应的变换系数值是正数；
- 否则（coeff_sign_flag 等于 1），对应的变换系数值是负值；

coeffLevel contains maxNumCoeff transform coefficient levels for the current list of transform coefficient levels.

CoeffLevel 包含了当前变换系数值列表中的 maxNumCoeff 个变换系数值。

8 Decoding process

Outputs of this process are decoded samples of the current picture (sometimes referred to by the variable CurrPic).

This clause describes the decoding process, given syntax elements and upper-case variables from clause 7.

The decoding process is specified such that all decoders shall produce numerically identical results. Any decoding process that produces identical results to the process described here conforms to the decoding process requirements of this Recommendation | International Standard.

Each picture referred to in this clause is a primary picture. Each slice referred to in this clause is a slice of a primary picture. Each slice data partition referred to in this clause is a slice data partition of a primary picture.

An overview of the decoding process is given as follows.

- The decoding of NAL units is specified in subclause 8.1.
- The processes in subclause 8.2 specify decoding processes using syntax elements in the slice layer and above.
  - Variables and functions relating to picture order count are derived in subclause 8.2.1. (only needed to be invoked for one slice of a picture)
  - Variables and functions relating to the macroblock to slice group map are derived in subclause 8.2.2. (only needed to be invoked for one slice of a picture)
  - The method of combining the various partitions when slice data partitioning is used is described in subclause 8.2.3.
  - Prior to decoding each slice, the derivation of reference picture lists as described in 8.2.4 is necessary for inter prediction.
  - When the current picture is a reference picture and after all slices of the current picture have been decoded, the decoded reference picture marking process in subclause 8.2.5 specifies how the current picture is used in the decoding process of inter prediction in later decoded pictures.
- The processes in subclauses 8.3, 8.4, 8.5, 8.6, and 8.7 specify decoding processes using syntax elements in the macroblock layer and above.
  - The intra prediction process for I and SI macroblocks except for I_PCM macroblocks as specified in subclause 8.3 provides the intra prediction samples being the output. For I_PCM macroblocks subclause 8.3 directly

specifies a picture construction process. The output are the constructed samples prior to the deblocking filter process.

- The inter prediction process for P and B macroblocks is specified in subclause 8.4 with inter prediction samples being the output.
- The decoding process transform coefficient and picture construction prior to deblocking filter process are specified in subclause 8.5. The transform coefficient decoding process derives the residual samples for I and B macroblocks as well as for P macroblocks in P slices. The output are the constructed samples prior to the deblocking filter process.
- The decoding process for transform coefficients and picture construction prior to deblocking for P macroblocks in SP slices or SI macroblocks is specified in subclause 8.6. The output are the constructed samples prior to the deblocking filter process.
- The constructed samples prior to the deblocking filter process that are next to the edges of blocks and macroblocks are processed by a deblocking filter as specified in subclause 8.7 with the output being the decoded samples.

8.1 NAL unit decoding process

Inputs to this process are NAL units.

Outputs of this process are the RBSP syntax structures encapsulated within the NAL units.

The decoding process for each NAL unit extracts the RBSP syntax structure from the NAL unit and then operates the decoding processes specified for the RBSP syntax structure in the NAL unit as follows.

Subclause 8.2 describes the decoding process for NAL units with nal_unit_type equal to 1 through 5.

Subclauses 8.3 describes the decoding process for a macroblock or part of a macroblock coded in NAL units with nal_unit_type equal to 1, 2, and 5.

Subclause 8.4 describes the decoding process for a macroblock or part of a macroblock coded in NAL units with nal_unit_type equal to 1 and 2.

Subclause 8.5 describes the decoding process for a macroblock or part of a macroblock coded in NAL units with nal_unit_type equal to 1 and 3 to 5.

Subclause 8.6 describes the decoding process for a macroblock or part of a macroblock coded in NAL units with nal_unit_type equal to 1 and 3 to 5.

Subclause 8.7 describes the decoding process for a macroblock or part of a macroblock coded in NAL units with nal_unit_type equal to 1 to 5.

NAL units with nal_unit_type equal to 7 and 8 contain sequence parameter sets and picture parameter sets, respectively. Picture parameter sets are used in the decoding processes of other NAL units as determined by reference to a picture parameter set within the slice headers of each picture. Sequence parameter sets are used in the decoding processes of other NAL units as determined by reference to a sequence parameter set within the picture parameter sets of each sequence.

No normative decoding process is specified for NAL units with nal_unit_type equal to 6, 9, 10, 11, and 12.

8.2 Slice decoding process

8.2.1 Decoding process for picture order count

8.2.1 图像顺序号的解码过程

Outputs of this process are TopFieldOrderCnt (if applicable) and BottomFieldOrderCnt (if applicable).

这个过程的输出是：TopFieldOrderCnt 和 BottomFieldOrderCnt

Picture order counts are used to determine initial picture orderings for reference pictures in the decoding of B slices (see subclauses 8.2.4.2.3 and 8.2.4.2.4), to represent picture order differences between frames or fields for motion vector derivation in temporal direct mode (see subclause 8.4.1.2.3), for implicit mode weighted prediction in B slices (see subclause 8.4.2.3.2), and for decoder conformance checking (see subclause C.4).

在解码 B slice的过程中，图像顺序号被用来确定参考图像的初始顺序号；在时域直接模式中计算运动矢量时、在 B slice 中隐含的加权预测模式中和解码器的兼容性测试时，被用来表示帧或场之间的图像顺序号差。

Picture order count information is derived for every frame, field (whether decoded from a coded field or as a part of a decoded frame), or complementary field pair as follows:

帧、场或互补场对的图像顺序号信息计算如下：

- Each coded frame is associated with two picture order counts, called TopFieldOrderCnt and BottomFieldOrderCnt for its top field and bottom field, respectively.
- 每个编码帧和两个图像序列号相关，分别对应于顶场和底场的 TopFieldOrderCnt 和 BottomFieldOrderCnt。
- Each coded field is associated with a picture order count, called TopFieldOrderCnt for a coded top field and BottomFieldOrderCnt for a bottom field.
- 每个编码场和一个图像序列号相关，如果编码的是顶场，则为 TopFieldOrderCnt；如果编码的是底场，则为 BottomFieldOrderCn。
- Each complementary field pair is associated with two picture order counts, which are the TopFieldOrderCnt for its coded top field and the BottomFieldOrderCnt for its coded bottom field, respectively.
- 互补场对也和两个图像序列号有关，分别对应于顶场和底场的 TopFieldOrderCnt 和 BottomFieldOrderCnt。

TopFieldOrderCnt and BottomFieldOrderCnt indicate the picture order of the corresponding top field or bottom field relative to the first output field of the previous IDR picture or the previous reference picture including a memory_management_control_operation equal to 5 in decoding order.

TopFieldOrderCnt 和 BottomFieldOrderCnt 分别表示相对于前一 IDR 帧或者在解码顺序上 memory_management_control_operation 等于 5 的前一参考图像的第一个输出场的顶场或底场的图像序列号。(注：memory_management_control_operation等于 5 表示重置所有的参考图像)

TopFieldOrderCnt and BottomFieldOrderCnt are derived by invoking one of the decoding processes for picture order count type 0, 1, and 2 in subclauses 8.2.1.1, 8.2.1.2, and 8.2.1.3, respectively. When the current picture includes a memory management control operation equal to 5, after the decoding of the current picture, tempPicOrderCnt is set equal to PicOrderCnt(CurrPic), TopFieldOrderCnt of the current picture (if any) is set equal to TopFieldOrderCnt - tempPicOrderCnt, and BottomFieldOrderCnt of the current picture (if any) is set equal to BottomFieldOrderCnt - tempPicOrderCnt.

TopFieldOrderCnt 和 BottomFieldOrderCnt的计算主要使用 8.2.1.1, 8.2.1.2, and 8.2.1.3 小节中的图像序列号类型分别为 1、2 和 3 中的一种。当当前图像的 memory_management_control_operation等于 5 ，那么解完当前图像之后，tempPicOrderCnt 被设置成 PicOrderCnt(CurrPic)，当前图像的 TopFieldOrderCnt（如果有的话）被设置成 TopFieldOrderCnt – tempPicOrderCnt ，而当前图像的 BottomFieldOrderCnt（如果有的话）被设置成 BottomFieldOrderCnt - tempPicOrderCnt。

The bitstream shall not contain data that results in Min(TopFieldOrderCnt, BottomFieldOrderCnt) not equal to 0 for a coded IDR frame, TopFieldOrderCnt not equal to 0 for a coded IDR top field, or BottomFieldOrderCnt not equal to 0 for a coded IDR bottom field. Thus, at least one of TopFieldOrderCnt and BottomFieldOrderCnt shall be equal to 0 for the fields of a coded IDR frame.

对于一帧编码 IDR 图像、一 IDR 顶场或一 IDR 底场来说，比特流中不应该包含Min(TopFieldOrderCnt, BottomFieldOrderCnt)不等于 0、TopFieldOrderCnt不等于 0 或者 BottomFieldOrderCnt 不等于 0，因此一IDR编码帧中 TopFieldOrderCnt 和 BottomFieldOrderCnt 至少有一个为 0 。

When the current picture is not an IDR picture, the following applies.

当当前图像不是 IDR图像时，则如下：

- Consider the list variable listD containing as elements the TopFieldOrderCnt and BottomFieldOrderCnt values associated with the list of pictures including all of the following
  - the first picture in the list is the previous picture of any of the following types
    - an IDR picture
    - a picture containing a memory_management_control_operation equal to 5
  - all other pictures that follow in decoding order after the first picture in the list and precede through the current picture which is also included in listD prior to the invoking of the decoded reference picture marking process.
- 假设列表变量 listD 包括和下面所列图像相关的 TopFieldOrderCnt 和 BottomFieldOrderCn 值
  - 列表中的第一帧图像下面所列类型中前一帧图像

- IDR 图像
- memory_management_control_operation 等于 5 的图像

- 列表 listD中还包括按解码顺序在当前图像帧之前、第一帧图像之后而且还没有进行解码参考图像标识处理的所有其他图像。

- Consider the list variable listO which contains the elements of listD sorted in ascending order. listO shall not contain any of the following.
  - a pair of TopFieldOrderCnt and BottomFieldOrderCnt for a frame or complementary field pair that are not at consecutive positions in listO.
  - a TopFieldOrderCnt that has a value equal to another TopFieldOrderCnt.
  - a BottomFieldOrderCnt that has a value equal to another BottomFieldOrderCnt.
  - a BottomFieldOrderCnt that has a value equal to a TopFieldOrderCnt unless the BottomFieldOrderCnt and TopFieldOrderCnt belong to the same coded frame or complementary field pair.
- 假设列表变量 listO 中包含按升排列的 listD 元素。但是 listO 中不包含下列情况：
  - 一帧或一互补场对的 TopFieldOrderCnt 和 BottomFieldOrderCnt 在 listO 中位置不连续；
  - 一图像的 TopFieldOrderCnt 等于另一图像 TopFieldOrderCnt；
  - 一图像的 BottomFieldOrderCn 等于另一图像 BottomFieldOrderCn；
  - 一图像的 BottomFieldOrderCn 等于另一图像 TopFieldOrderCnt，除非 BottomFieldOrderCn 和 TopFieldOrderCnt 属于相同的编码帧或者相同的互补场对；

The bitstream shall not contain data that results in values of TopFieldOrderCnt, BottomFieldOrderCnt, PicOrderCntMsb, or FrameNumOffset used in the decoding process as specified in subclauses 8.2.1.1 to 8.2.1.3 that exceed the range of values from -2^{31} to $2^{31}-1$, inclusive.

比特流中不应该这样的数据，使在象 8.2.1.1 到 8.2.1.3 小节的解码过程中的 TopFieldOrderCnt, BottomFieldOrderCnt, PicOrderCntMsb, 或者 FrameNumOffset的值超出[-2^{31} $2^{31}-1$]的范围。

The function PicOrderCnt(picX) is specified as follows:

```
if( picX is a frame or a complementary field pair )
   PicOrderCnt( picX ) = Min( TopFieldOrderCnt, BottomFieldOrderCnt ) of complementary field pair picX
else if( picX is a top field )
   PicOrderCnt( picX ) = TopFieldOrderCnt of field picX                                          (8-1)
else if( picX is a bottom field )
   PicOrderCnt( picX ) = BottomFieldOrderCnt of field picX
```

函数 PicOrderCnt(picX) 的说明如下：

if（picX 是一帧或一互补场对）

PicOrderCnt(picX) ＝互补场对 picX 的 TopFieldOrderCnt 和 BottomFieldOrderCnt中的最小值

if（picX 是一顶场）

PicOrderCnt(picX) ＝ 顶场的 TopFieldOrderCnt

if（picX 是一底场）

PicOrderCnt(picX) ＝ 底场的 TopFieldOrderCnt

Then DiffPicOrderCnt(picA, picB) is specified as follows:

DiffPicOrderCnt(picA, picB) = PicOrderCnt(picA) - PicOrderCnt(picB)   (8-2)

函数 DiffPicOrderCnt(picA, picB)的说明如下：

DiffPicOrderCnt(picA, picB) = PicOrderCnt(picA) - PicOrderCnt(picB)

The bitstream shall contain data that results in values of DiffPicOrderCnt(picA, picB) used in the decoding process that are in the range of -2^{15} to $2^{15} - 1$, inclusive.

码流中包含的数据应该保证解码过程中 DiffPicOrderCnt(picA, picB) 的值在[-2^{15}，$2^{15} - 1$]中。

NOTE – Let X be the current picture and Y and Z be two other pictures in the same sequence, Y and Z are considered to be in the same output order direction from X when both DiffPicOrderCnt(X, Y) and DiffPicOrderCnt(X, Z) are positive or both are negative.

NOTE – Many applications assign PicOrderCnt(X) proportional to the sampling time of the picture X relative to the sampling time of an IDR picture.

注一 假设 X 是当前图像，Y 和 Z 是相同序列中的其它两帧图像，当DiffPicOrderCnt(X, Y)和DiffPicOrderCnt(X, Z)同时为正或同时为负时，那么Y 和 Z 被认为相对于 X 有相同的输出顺序方向。

注一 许多应用认为 PicOrderCnt(X) 与 图像 X 相对于 IDR 图像 的采样时间成正比。

8.2.1.1 Decoding process for picture order count type 0

8.2.1.1 图像顺序号类型为 0 的解码过程

This process is invoked when pic_order_cnt_type is equal to 0.

当 pic_order_cnt_type 的值为 0 ，则使用本解码过程。

Input to this process is PicOrderCntMsb of the previous reference picture in decoding order as specified in this subclause.

本解码过程的输入是本小节所示的按照解码顺序的前一参考图像的 PicOrderCntMsb。

Outputs of this process are either or both TopFieldOrderCnt or BottomFieldOrderCnt.

本解码过程的输出是 TopFieldOrderCnt 和 BottomFieldOrderCnt 或者它们两者之一。

The variable prevPicOrderCntMsb is derived as follows.

- If the current picture is an IDR picture, prevPicOrderCntMsb is set equal to 0 and prevPicOrderCntLsb is set equal to 0.
- If the current picture is not an IDR picture and the previous decoded picture in decoding order included a memory_management_control_operation equal to 5, prevPicOrderCntMsb is set equal to 0 and prevPicOrderCntLsb is set equal to 0.
- Otherwise (the current picture is not an IDR picture and the previous decoded picture in decoding order included not a memory_management_control_operation equal to 5), prevPicOrderCntMsb is set equal to PicOrderCntMsb of the previous reference picture in decoding order and prevPicOrderCntLsb is set equal to the value of pic_order_cnt_lsb of the previous reference picture in decoding order.

变量 prevPicOrderCntMsb 的计算如下：

- 如果当前图像是一 IDR 图像，那么 prevPicOrderCntMsb 和 prevPicOrderCntLsb 都被设置成 0；
- 如果当前图像不是 IDR 图像，但是解码顺序上的前一解码图像的 memory_management_control_operation 的值为 5 ， 那么 prevPicOrderCntMsb 和 prevPicOrderCntLsb 都被设置成
- 否则（当前图像不是 IDR 图像，并且解码顺序上的前一解码图像的 memory_management_control_operation 的值也不为 5），prevPicOrderCntMsb 被设置成解码顺序上前一参考图像的 prevPicOrderCntMsb ，prevPicOrderCntLsb 被设置成 解码顺序上前一参考图像的 prevPicOrderCntLsb 。

PicOrderCntMsb of the current picture is derived as follows:

```
if( ( pic_order_cnt_lsb < prevPicOrderCntLsb ) &&
    ( ( prevPicOrderCntLsb – pic_order_cnt_lsb ) >= ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb                    (8-3)
else if( ( pic_order_cnt_lsb > prevPicOrderCntLsb ) &&
        ( ( pic_order_cnt_lsb – prevPicOrderCntLsb ) > ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb – MaxPicOrderCntLsb
else
    PicOrderCntMsb = prevPicOrderCntMsb
```

当前图像的 PicOrderCntMsb 计算如下(建议举个例子了解具体的过程，见JVT-F044)：

```
if( ( pic_order_cnt_lsb < prevPicOrderCntLsb ) &&
   ( ( prevPicOrderCntLsb – pic_order_cnt_lsb ) >= ( MaxPicOrderCntLsb / 2 ) ) )
  PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb
else if( ( pic_order_cnt_lsb > prevPicOrderCntLsb ) &&
       ( ( pic_order_cnt_lsb – prevPicOrderCntLsb ) > ( MaxPicOrderCntLsb / 2 ) ) )
  PicOrderCntMsb = prevPicOrderCntMsb – MaxPicOrderCntLsb
else
  PicOrderCntMsb = prevPicOrderCntMsb
```

When the current picture is not a bottom field, TopFieldOrderCnt is derived as follows:

```
if( !field_pic_flag | | !bottom_field_flag )
  TopFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb                    (8-4)
```

当当前图像不是底场时，TopFieldOrderCnt 的计算如下所示：

```
if( !field_pic_flag | | !bottom_field_flag )
    TopFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb
```

When the current picture is not a top field, BottomFieldOrderCnt is derived as follows:

```
if( !field_pic_flag )
  BottomFieldOrderCnt = TopFieldOrderCnt + delta_pic_order_cnt_bottom
else if( bottom_field_flag )                                               (8-5)
  BottomFieldOrderCnt =  PicOrderCntMsb + pic_order_cnt_lsb
```

当当前图像不是顶场时，BottomFieldOrderCnt 的计算如下所示：

```
if( !field_pic_flag )
    BottomFieldOrderCnt = TopFieldOrderCnt + delta_pic_order_cnt_bottom
else if( bottom_field_flag )
    BottomFieldOrderCnt =  PicOrderCntMsb + pic_order_cnt_lsb
```

8.2.1.2 Decoding process for picture order count type 1

8.2.1.2 图像顺序号类型为 1 的解码过程

This process is invoked when pic_order_cnt_type is equal to 1.

当 pic_order_cnt_type 的值为 0， 则使用本解码过程。

Input to this process is FrameNumOffset of the previous picture in decoding order as specified in this subclause.

本解码过程的输入是本小节所示的按照解码顺序的前一图像的 FrameNumOffset。

Outputs of this process are either or both TopFieldOrderCnt or BottomFieldOrderCnt.

本解码过程的输出是 TopFieldOrderCnt 和 BottomFieldOrderCnt 或者它们两者之一。

The values of TopFieldOrderCnt and BottomFieldOrderCnt are derived as specified in this subclause. Let prevFrameNum be equal to the frame_num of the previous picture in decoding order.

TopFieldOrderCnt 和 BottomFieldOrderCnt 值的计算如本小节所示，假设 prevFrameNum 为解码顺序上前一图像的 frame_num

When the current picture is not an IDR picture, the variable prevFrameNumOffset is derived as follows.

- If the previous picture in decoding order included a memory_management_control_operation equal to 5, prevFrameNumOffset is set equal to 0.
- Otherwise (the previous picture in decoding order did not include a memory_management_control_operation equal to 5), prevFrameNumOffset is set equal to the value of FrameNumOffset of the previous picture.

当当前图像不是 IDR 图像，那么变量 prevFrameNumOffset 的计算如下：

- 如果解码顺序上前一图像的 memory_management_control_operation 的值为 5 ， 那么 prevFrameNumOffset的值设置为 0 ；

- 否则（解码顺序上前一图像的 memory_management_control_operation 的值不为 5），prevFrameNumOffset 的值被设置为前一图像的 FrameNumOffset 。

The derivation proceeds in the following ordered steps.

计算步骤按下面的步骤进行

1. The variable FrameNumOffset is derived as follows:

   变量 FrameNumOffset 的计算如下：

```
if( nal_unit_type == 5 )
    FrameNumOffset = 0
else if( prevFrameNum > frame_num )                                        (8-6)
    FrameNumOffset = prevFrameNumOffset + MaxFrameNum
else
    FrameNumOffset = prevFrameNumOffset
```

2. The variable absFrameNum is derived as follows:

   变量 absFrameNum 计算如下：

```
if( num_ref_frames_in_pic_order_cnt_cycle != 0 )
    absFrameNum = FrameNumOffset + frame_num
else                                                                       (8-7)
    absFrameNum = 0
if( nal_ref_idc == 0 && absFrameNum > 0 )
    absFrameNum = absFrameNum – 1
```

3. When absFrameNum > 0, picOrderCntCycleCnt and frameNumInPicOrderCntCycle are derived as follows:

   当 absFrameNum > 0 时， picOrderCntCycleCnt 和 frameNumInPicOrderCntCycle 的计算如下：

```
if( absFrameNum > 0 ) {
    picOrderCntCycleCnt = ( absFrameNum – 1 ) / num_ref_frames_in_pic_order_cnt_cycle
    frameNumInPicOrderCntCycle = ( absFrameNum – 1 ) % num_ref_frames_in_pic_order_cnt_cycle    (8-8)
}
```

4. The variable expectedDeltaPerPicOrderCntCycle is derived as follows:

   变量 expectedDeltaPerPicOrderCntCycle 的计算如下：

```
expectedDeltaPerPicOrderCntCycle = 0
for( i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++ )
    expectedDeltaPerPicOrderCntCycle += offset_for_ref_frame[ i ]              (8-9)
```

5. The variable expectedPicOrderCnt is derived as follows:

   变量 expectedPicOrderCnt 的计算如下：

```
if( absFrameNum > 0 ){
    expectedPicOrderCnt = picOrderCntCycleCnt * expectedDeltaPerPicOrderCntCycle
    for( i = 0; i <= frameNumInPicOrderCntCycle; i++ )
        expectedPicOrderCnt = expectedPicOrderCnt + offset_for_ref_frame[ i ]
} else
    expectedPicOrderCnt = 0
if( nal_ref_idc == 0 )                                                     (8-10)
    expectedPicOrderCnt = expectedPicOrderCnt + offset_for_non_ref_pic
```

6. The variables TopFieldOrderCnt or BottomFieldOrderCnt are derived as follows:

   变量 TopFieldOrderCnt 或 BottomFieldOrderCnt 的计算如下：

```
if( !field_pic_flag ) {
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
    BottomFieldOrderCnt = TopFieldOrderCnt +
```

```
            offset_for_top_to_bottom_field + delta_pic_order_cnt[ 1 ]                    (8-11)
} else if( !bottom_field_flag )
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
else
    BottomFieldOrderCnt = expectedPicOrderCnt + offset_for_top_to_bottom_field + delta_pic_order_cnt[ 0 ]
```

8.2.1.3 Decoding process for picture order count type 2

8.2.1.3 图像顺序号类型为 3 的解码过程

This process is invoked when pic_order_cnt_type is equal to 2.

当 pic_order_cnt_type 的值为 3 ，则使用本解码过程。

Outputs of this process are either or both TopFieldOrderCnt or BottomFieldOrderCnt.

本解码过程的输出是 TopFieldOrderCnt 和 BottomFieldOrderCnt 或者它们两者之一。

Let prevFrameNum be equal to the frame_num of the previous picture in decoding order.

假设 prevFrameNum 为按解码顺序的前一图像的 frame_num。

When the current picture is not an IDR picture, the variable prevFrameNumOffset is derived as follows.

- If the previous picture in decoding order included a memory_management_control_operation equal to 5, prevFrameNumOffset is set equal to 0.
- Otherwise (the previous picture in decoding order did not include a memory_management_control_operation equal to 5), prevFrameNumOffset is set equal to the value of FrameNumOffset of the previous picture.

当当前图像不是 IDR 图像，那么变量 prevFrameNumOffset 的计算如下：

- 如果按解码顺序前一图像的 memory_management_control_operation 等于 5 ，那么 prevFrameNumOffset 的值设置为 0 。
- 否则（按解码顺序前一图像的 memory_management_control_operation 不等于 5），prevFrameNumOffset 的值设置为前一图像的 FrameNumOffset 。

The variable FrameNumOffset is derived as follows.

变量 FrameNumOffset 的计算如下：

```
if( nal_unit_type  = =  5 )
    FrameNumOffset = 0
else if( prevFrameNum  >  frame_num )                                             (8-12)
    FrameNumOffset = prevFrameNumOffset + MaxFrameNum
else
    FrameNumOffset = prevFrameNumOffset
```

The variable tempPicOrderCnt is derived as follows:

变量 tempPicOrderCnt 的计算如下：

```
if( nal_unit_type  = =  5 )
    tempPicOrderCnt = 0
else if( nal_ref_idc  = =  0 )                                                    (8-13)
    tempPicOrderCnt = 2 * (FrameNumOffset + frame_num) – 1
else
    tempPicOrderCnt = 2 * (FrameNumOffset + frame_num)
```

The variables TopFieldOrderCnt or BottomFieldOrderCnt are derived as follows:

变量 TopFieldOrderCnt 或 BottomFieldOrderCnt的计算如下：

```
if( !field_pic_flag ) {
    TopFieldOrderCnt = tempPicOrderCnt
    BottomFieldOrderCnt = tempPicOrderCnt                                         (8-14)
} else if( bottom_field_flag )
    BottomFieldOrderCnt = tempPicOrderCnt
```

else
    TopFieldOrderCnt = tempPicOrderCnt

NOTE – Picture order count type 2 cannot be used in a sequence that contains consecutive non-reference pictures of the form specified in subclause 7.4.2.1.

NOTE –Picture order count type 2 results in an output order that is the same as the decoding order.

注 － 在 7.4.2.1 所述的连续非参考图像中不能使用 图像顺序号类型为 2 的图像顺序号编码方式

注 － 图像顺序号类型为 2 时的输出顺序和解码顺序相同。

8.2.2 Decoding process for macroblock to slice group map

8.2.2 宏块到 slice 组的映射解码处理（对不同的类型最好给出图图形化的表示，如JVT-C089）

Inputs to this process are the active picture parameter set and the slice header of the slice to be decoded.

Output of this process is a macroblock to slice group map MbToSliceGroupMap.

本过程的输入是当前的图像参数集和当前解码的 slice 的头信息。

本过程的输出是宏块到 slice组的映射 MbToSliceGroupMap。

This process is invoked at the start of every slice.

NOTE – The output of this process is equal for all slices of an access unit.

本过程在每一个 slice的开始解码时调用。

注－ 本过程的输出对一个访问单元的所有 slice 都是相同的。

When num_slice_groups_minus1 is equal to 1 and slice_group_map_type is equal to 3, 4, or 5, slice groups 0 and 1 have a size and shape determined by slice_group_change_direction_flag as shown in Table 8-1 and specified in subclauses 8.2.2.4-8.2.2.6.

Table 8-1 – Refined slice group map type

slice_group_map_type	slice_group_change_direction_flag	refined slice group map type
3	0	Box-out clockwise
3	1	Box-out counter-clockwise
4	0	Raster scan
4	1	Reverse raster scan
5	0	Wipe right
5	1	Wipe left

当 num_slice_groups_minus1等于 1，并且 slice_group_map_type 等于 3、4 或 5 时，slice 组 0 和 slice 组 1的大小和形状表 8－1中的 slice_group_change_direction_flag 来确定，如 8.2.2.4-8.2.2.6 小节所述。

In such a case, MapUnitsInSliceGroup0 slice group map units in the specified growth order are allocated for slice group 0 and the remaining PicSizeInMapUnits – MapUnitsInSliceGroup0 slice group map units of the picture are allocated for slice group 1.

在这种情况下，MapUnitsInSliceGroup0 个 slice 组映射单元按说明的扩展顺序分配给 slice组 0，剩余的 PicSizeInMapUnits – MapUnitsInSliceGroup0 个 slice 组映射单元分配给 slice 组 1。

When num_slice_groups_minus1 is equal to 1 and slice_group_map_type is equal to 4 or 5, the variable sizeOfUpperLeftGroup is defined as follows:

sizeOfUpperLeftGroup = (slice_group_change_direction_flag ?
    (PicSizeInMapUnits – MapUnitsInSliceGroup0) : MapUnitsInSliceGroup0)   (8-15)

当 num_slice_groups_minus1等于 1，并且 slice_group_map_type 等~~于 3、~~4 或 5 时，变量 sizeOfUpperLeftGroup的定义如下：

sizeOfUpperLeftGroup = (slice_group_change_direction_flag ?
    (PicSizeInMapUnits – MapUnitsInSliceGroup0) : MapUnitsInSliceGroup0)

The variable mapUnitToSliceGroupMap is derived as follows.

- If num_slice_groups_minus1 is equal to 0, the map unit to slice group map is generated for all i ranging from 0 to PicSizeInMapUnits – 1, inclusive, as specified by:

  mapUnitToSliceGroupMap[i] = 0 (8-16)

- Otherwise (num_slice_groups_minus1 is not equal to 0), mapUnitToSliceGroupMap is derived as follows.
  – If slice_group_map_type is equal to 0, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.1 applies.
  – If slice_group_map_type is equal to 1, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.2 applies.
  – If slice_group_map_type is equal to 2, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.3 applies.
  – If slice_group_map_type is equal to 3, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.4 applies.
  – If slice_group_map_type is equal to 4, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.5 applies.
  – If slice_group_map_type is equal to 5, the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.6 applies.
  – Otherwise (slice_group_map_type is equal to 6), the derivation of mapUnitToSliceGroupMap as specified in subclause 8.2.2.7 applies.

变量 mapUnitToSliceGroupMap的计算如下：

- 如果 num_slice_groups_minus1 等于 0，那么映射单元到 slice组的映射按下述的方法产生， i 的范围是[0, PicSizeInMapUnits – 1]

  mapUnitToSliceGroupMap[i] = 0

- 否则（num_slice_groups_minus1 不等于 0），mapUnitToSliceGroupMap的计算如下：
  – 如果 slice_group_map_type 等于 0， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.1 所示；
  – 如果 slice_group_map_type 等于 1， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.2 所示；
  – 如果 slice_group_map_type 等于 2， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.3 所示；
  – 如果 slice_group_map_type 等于 3， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.4 所示；
  – 如果 slice_group_map_type 等于 4， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.5 所示；
  – 如果 slice_group_map_type 等于 5， 那么 mapUnitToSliceGroupMap 的计算如 8.2.2.6 所示；
  – 否则（slice_group_map_type 等于 6）， mapUnitToSliceGroupMap 的计算如 8.2.2.7所示。

After derivation of the mapUnitToSliceGroupMap, the process specified in subclause 8.2.2.8 is invoked to convert the map unit to slice group map mapUnitToSliceGroupMap to the macroblock to slice group map MbToSliceGroupMap. After derivation of the macroblock to slice group map as specified in subclause 8.2.2.8, the function NextMbAddress(n) is defined as the value of the variable nextMbAddress derived as specified by:

```
i = n + 1
while( i < PicSizeInMbs  &&  MbToSliceGroupMap[ i ]  !=  MbToSliceGroupMap[ n ] )
    i++;
nextMbAddress = i                                                        (8-17)
```

完成 mapUnitToSliceGroupMap的计算之后，再使用 8.2.2.8所示的方法将映射到 slice组 mapUnitToSliceGroupMap的映射单元转化成映射到 slice组 MbToSliceGroupMap的宏块。函数 NextMbAddress(n)被定义为计算变量 nextMbAddress的值，如下所示：

```
i = n + 1
while( i < PicSizeInMbs  &&  MbToSliceGroupMap[ i ]  !=  MbToSliceGroupMap[ n ] )
  i++;
nextMbAddress = i
```

8.2.2.1 Specification for interleaved slice group map type

8.2.2.1 交叉型 slice组映射类型的描述（隔行扫描的翻译好像有问题）

The specifications in this subclause apply when slice_group_map_type is equal to 0.

当 slice_group_map_type等于 0 时使用本小节的描述

The map unit to slice group map is generated as specified by:

```
i = 0
do
    for( iGroup = 0; iGroup <= num_slice_groups_minus1 && i < PicSizeInMapUnits;
         i += run_length_minus1[ iGroup++ ] + 1 )
        for( j = 0; j <= run_length_minus1[ iGroup ] && i + j < PicSizeInMapUnits; j++ )
            mapUnitToSliceGroupMap[ i + j ] = iGroup                                   (8-18)
while( i < PicSizeInMapUnits )
```

映射到 slice 组的映射单元的产生方法如下：

```
i = 0
do
    for( iGroup = 0; iGroup <= num_slice_groups_minus1 && i < PicSizeInMapUnits;
         i += run_length_minus1[ iGroup++ ] + 1 )
        for( j = 0; j <= run_length_minus1[ iGroup ] && i + j < PicSizeInMapUnits; j++ )
            mapUnitToSliceGroupMap[ i + j ] = iGroup
while( i < PicSizeInMapUnits )
```

8.2.2.2 Specification for dispersed slice group map type

8.2.2.2 分散型 slice 组映射类型的描述

The specifications in this subclause apply when slice_group_map_type is equal to 1.

当 slice_group_map_type等于 1 时使用本小节的描述

The map unit to slice group map is generated as specified by:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = ( ( i % PicWidthInMbs ) +
                                  ( ( ( i / PicWidthInMbs ) * ( num_slice_groups_minus1 + 1 ) ) / 2 ) )
                                  % ( num_slice_groups_minus1 + 1 )                 (8-19)
```

映射到 slice 组的映射单元的产生方法如下：

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = ( ( i % PicWidthInMbs ) +
                                  ( ( ( i / PicWidthInMbs ) * ( num_slice_groups_minus1 + 1 ) ) / 2 ) )
                                  % ( num_slice_groups_minus1 + 1 )
```

8.2.2.3 Specification for foreground with left-over slice group map type

8.2.2.3 前景加剩余型 slice 组映射类型的描述（名称有待参考）

The specifications in this subclause apply when slice_group_map_type is equal to 2.

当 slice_group_map_type等于 2 时使用本小节的描述

The map unit to slice group map is generated as specified by:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = num_slice_groups_minus1
for( iGroup = num_slice_groups_minus1 – 1; iGroup >= 0; iGroup-- ) {
    yTopLeft = top_left[ iGroup ] / PicWidthInMbs
    xTopLeft = top_left[ iGroup ] % PicWidthInMbs
    yBottomRight = bottom_right[ iGroup ] / PicWidthInMbs
    xBottomRight = bottom_right[ iGroup ] % PicWidthInMbs
```

```
        for( y = yTopLeft; y <= yBottomRight; y++ )
            for( x = xTopLeft; x <= xBottomRight; x++ )
                mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = iGroup        (8-20)
    }
```

映射到 slice 组的映射单元的产生方法如下：

```
    for( i = 0; i < PicSizeInMapUnits; i++ )
        mapUnitToSliceGroupMap[ i ] = num_slice_groups_minus1
    for( iGroup = num_slice_groups_minus1 – 1; iGroup >= 0; iGroup-- ) {
        yTopLeft = top_left[ iGroup ] / PicWidthInMbs
        xTopLeft = top_left[ iGroup ] % PicWidthInMbs
        yBottomRight = bottom_right[ iGroup ] / PicWidthInMbs
        xBottomRight = bottom_right[ iGroup ] % PicWidthInMbs
        for( y = yTopLeft; y <= yBottomRight; y++ )
            for( x = xTopLeft; x <= xBottomRight; x++ )
                mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = iGroup
    }
```

After application of the process specified in Equation 8-20, there shall be at least one value of i from 0 to PicSizeInMapUnits – 1, inclusive, for which mapUnitToSliceGroupMap[i] is equal to iGroup for each value of iGroup from 0 to num_slice_groups_minus1, inclusive (i.e., each slice group shall contain at least one slice group map unit).

在上述等式 8-20 处理完之后，至少有一个值 i，i的范围是[0，PicSizeInMapUnits – 1]，使得 mapUnitToSliceGroupMap[i]等于 iGroup，iGroup的范围是[0，num_slice_groups_minus1]（例如：每个 slice组至少包含一个 slice组映射单元）。

NOTE – The rectangles may overlap. Slice group 0 contains the macroblocks that are within the rectangle specified by top_left[0] and bottom_right[0]. A slice group having slice group ID greater than 0 and less than num_slice_groups_minus1 contains the macroblocks that are within the specified rectangle for that slice group that are not within the rectangle specified for any slice group having a smaller slice group ID. The slice group with slice group ID equal to num_slice_groups_minus1 contains the macroblocks that are not in the other slice groups.

注 – 矩形也许会重叠，slice组 0 中包含有由 top_left[0] 和 bottom_right[0]规定的矩形中的宏块。Slice 组 ID 大于 0 并且小于 num_slice_groups_minus1 的 slice 组包含有规定矩形中的宏块，但是不包含由任何更小的 slice 组 ID 已经包含的宏块。Slice 组 ID等于 num_slice_groups_minus1 contains 的 slice 组包括了其他 slice组没有包含的宏块。（最好用图示说明）

8.2.2.4 Specification for box-out slice group map types

8.2.2.4 外旋盒子型 slice 组映射类型的描述（名称有待参考）

The specifications in this subclause apply when slice_group_map_type is equal to 3.

当 slice_group_map_type等于 3 时使用本小节的描述

The map unit to slice group map is generated as specified by:

映射到 slice 组的映射单元的产生方法如下：

```
    for( i = 0; i < PicSizeInMapUnits; i++ )
        mapUnitToSliceGroupMap[ i ] = 1
    x = ( PicWidthInMbs – slice_group_change_direction_flag ) / 2
    y = ( PicHeightInMapUnits – slice_group_change_direction_flag ) / 2
    ( leftBound, topBound ) = ( x, y )
    ( rightBound, bottomBound ) = ( x, y )
    ( xDir, yDir ) = ( slice_group_change_direction_flag – 1, slice_group_change_direction_flag )
    for( k = 0; k < MapUnitsInSliceGroup0; k += mapUnitVacant ) {
        mapUnitVacant = ( mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = = 1 )
        if( mapUnitVacant )
            mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = 0                 (8-21)
        if( xDir = = –1 && x = = leftBound ) {
            leftBound = Max( leftBound – 1, 0 )
            x = leftBound
            ( xDir, yDir ) = ( 0, 2 * slice_group_change_direction_flag – 1 )
        } else if( xDir = = 1 && x = = rightBound ) {
            rightBound = Min( rightBound + 1, PicWidthInMbs – 1 )
            x = rightBound
```

```
        ( xDir, yDir ) = ( 0, 1 – 2 * slice_group_change_direction_flag )
    } else if( yDir  = =  –1  &&  y  = =  topBound ) {
        topBound = Max( topBound – 1, 0 )
        y = topBound
        ( xDir, yDir ) = ( 1 – 2 * slice_group_change_direction_flag, 0 )
    } else if( yDir  = =  1  &&  y  = =  bottomBound ) {
        bottomBound  =  Min( bottomBound + 1, PicHeightInMapUnits – 1 )
        y = bottomBound
        ( xDir, yDir ) = ( 2 * slice_group_change_direction_flag – 1, 0 )
    } else
        ( x, y ) = ( x + xDir, y + yDir )
}
```

8.2.2.5 Specification for raster scan slice group map types

8.2.2.5 栅格扫描型 slice 组映射类型的描述（名称有待参考）

The specifications in this subclause apply when slice_group_map_type is equal to 4.

当 slice_group_map_type等于 4 时使用本小节的描述

The map unit to slice group map is generated as specified by:

映射到 slice 组的映射单元的产生方法如下：

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    if( i < sizeOfUpperLeftGroup )
        mapUnitToSliceGroupMap[ i ] = slice_group_change_direction_flag
    else                                                                        (8-22)
        mapUnitToSliceGroupMap[ i ] = 1 – slice_group_change_direction_flag
```

8.2.2.6 Specification for wipe slice group map types

8.2.2.6 擦除型 slice 组映射类型的描述（名称有待参考）

The specifications in this subclause apply when slice_group_map_type is equal to 5.

当 slice_group_map_type等于 5 时使用本小节的描述

The map unit to slice group map is generated as specified by:

映射到 slice 组的映射单元的产生方法如下：

```
k = 0;
for( j = 0; j < PicWidthInMbs; j++ )
    for( i = 0; i < PicHeightInMapUnits; i++ )
        if( k++ < sizeOfUpperLeftGroup )
            mapUnitToSliceGroupMap[ i * PicWidthInMbs + j ] = slice_group_change_direction_flag
        else                                                                    (8-23)
            mapUnitToSliceGroupMap[ i * PicWidthInMbs + j ] = 1 – slice_group_change_direction_flag
```

8.2.2.7 Specification for explicit slice group map type

8.2.2.6 显式型 slice 组映射类型的描述（名称有待参考）

The specifications in this subclause apply when slice_group_map_type is equal to 6.

当 slice_group_map_type等于 6 时使用本小节的描述

The map unit to slice group map is generated as specified by:

映射到 slice 组的映射单元的产生方法如下：

mapUnitToSliceGroupMap[i] = slice_group_id[i]   (8-24)

for all i ranging from 0 to PicSizeInMapUnits – 1, inclusive.

i 的范围是[0，PicSizeInMapUnits – 1]

8.2.2.8 Specification for conversion of map unit to slice group map to macroblock to slice group map

8.2.2.8 由映射到 slice组的映射单元到映射宏块转化的描述（名称还需要推敲）

The macroblock to slice group map is specified as follows for each value of i ranging from 0 to PicSizeInMbs – 1, inclusive.

由宏块到 slice 组的映射说明如下，i 的范围是[0，PicSizeInMapUnits – 1]

– If frame_mbs_only_flag is equal to 1 or field_pic_flag is equal to 1, the macroblock to slice group map is specified by:

– 如果 frame_mbs_only_flag 等于 1 或者 field_pic_flag 等于 1，那么宏块到 slice 组的映射说明如下：

$$\text{MbToSliceGroupMap}[\,i\,] = \text{mapUnitToSliceGroupMap}[\,i\,] \quad (8\text{-}25)$$

– If MbaffFrameFlag is equal to 1, the macroblock to slice group map is specified by:

– 如果 MbaffFrameFlag 等于 1，那么宏块到 slice 组的映射说明如下：

$$\text{MbToSliceGroupMap}[\,i\,] = \text{mapUnitToSliceGroupMap}[\,i / 2\,] \quad (8\text{-}26)$$

– Otherwise (frame_mbs_only_flag is equal to 0 and mb_adaptive_frame_field_flag is equal to 0 and field_pic_flag is equal to 0), the macroblock to slice group map is specified by:

– 否则（frame_mbs_only_flag 、mb_adaptive_frame_field_flag 和 field_pic_flag 都等于 0 的情况），宏块到 slice 组的映射说明如下：

$$\text{MbToSliceGroupMap}[\,i\,] = \text{mapUnitToSliceGroupMap}[\,(\,i / (\,2 * \text{PicWidthInMbs}\,)\,) * \text{PicWidthInMbs} + (\,i \,\%\, \text{PicWidthInMbs}\,)\,] \quad (8\text{-}27)$$

8.2.3 Decoding process for slice data partitioning

8.2.3 slice 层数据分割的解码处理过程

Inputs to this process are

– a slice data partition A layer RBSP,

– when syntax elements of category 3 are present in the slice data, a slice data partition B layer RBSP having the same slice_id as in the slice data partition A layer RBSP, and

– when syntax elements of category 4 are present in the slice data, a slice data partition C layer RBSP having the same slice_id as in the slice data partition A layer RBSP.

NOTE – The slice data partition B layer RBSP and slice data partition C layer RBSP need not be present.

这个处理过程的输入是：（名称需要推敲）

– 一 slice数据块 A 层 RBSP

– 当在 slice 数据中出现类别为 3 的语法元素时，和 slice 数据块 A 层 RBSP有相同的 slcie 数据块 B 层RBSP

– 当在 slice 数据中出现类别为 4 的语法元素时，和 slice 数据块 A 层 RBSP有相同的 slcie 数据块 C 层RBSP

注一 slice 数据块 B层 RBSP和 slice 数据块 C 层 RBSP 不一定出现在码流中。

Output of this process is a coded slice.

When slice data partitioning is not used, coded slices are represented by a slice layer without partitioning RBSP that contains a slice header followed by a slice data syntax structure that contains all the syntax elements of categories 2, 3, and 4 (see category column in subclause 7.3) of the macroblock data for the macroblocks of the slice.

本解码过程的输出是一编码的 slice。

当 slice 数据分割不可用时，那么用没有分割的 RBSP slice 层来表示编码的 slice，这种没有分割的 slice层包含有slice头信息，后面接着是由该 slice 中的所有宏块的类别为2、3和 4 的所有语法元素（见 7.3小节中的类别那一列）组成的 slice 数据语法结构。

When slice data partitioning is used, the macroblock data of a slice is partitioned into one to three partitions contained in separate NAL units. Partition A contains a slice data partition A header, and all syntax elements of category 2. Partition B, when present, contains a slice data partition B header and all syntax elements of category 3. Partition C, when present, contains a slice data partition C header and all syntax elements of category 4.

当 slice 数据分割可用时，该 slice 的宏块数据被分割成一到三个块，分别包含在单独的 NAL 单元中。块 A 包含 slice 数据块 A 的头信息和类别为 2 的所有语法元素。块 B ，如果有的话，包含 slice 数据块 B 的头信息和类别为 3 的所有语法元素。块 C ，如果有的话，包含 slice 数据块 C 的头信息和类别为 4 的所有语法元素。

When slice data partitioning is used, the syntax elements of each category are parsed from a separate NAL unit, which need not be present when no symbols of the respective category exist. The decoding process shall process the slice data partitions of a coded slice in a manner equivalent to processing a corresponding slice layer without partitioning RBSP by extracting each syntax element from the slice data partition in which the syntax element appears depending on the slice data partition assignment in the syntax tables in subclause 7.3.

当 slice数据分割可用时，从单个的 NAL 单元中解析各个类别的语法元素，如果没有对应的语法元素，也就没有对应的 NAL 单元。对 slice 数据块的解码过程和对应的没有分割 RBSP的 slice 层的解码过程相同，除了将每一个语法元素从对应的 slice数据块中抽取出来。

NOTE - Syntax elements of category 3 are relevant to the decoding of residual data of I and SI macroblock types. Syntax elements of category 4 are relevant to the decoding of residual data of P and B macroblock types. Category 2 encompasses all other syntax elements related to the decoding of macroblocks, and their information is often denoted as header information. The slice data partition A header contains all the syntax elements of the slice header, and additionally a slice_id that are used to associate the slice data partitions B and C with the slice data partition A. The slice data partition B and C headers contain only the slice_id that establishes their association with the slice data partition A of the slice.

注一 类别为 3 的语法元素和 I 与 SI 类型的宏块的残差数据的解码有关。类别为 4 的语法元素和 P 与 B 宏块类型的残差数据的解码有关，类别 2 则包含了所有和宏块解码相关的其他语法元素，并且这些信息经常被称为头信息。Slice 数据块 A的头包含了 slice 头的所有语法元素，并且有一个 slice_id 用来使 slice 数据块 B 和 C 与 slice 数据块 A 相关联。而 slice 数据块 B和 C 的头仅仅包含了 slice_id，用来和该 slice 的数据块 A 相关联。

8.2.4 Decoding process for reference picture lists construction

8.2.4 参考图像列表构造的解码过程

This process is invoked at the beginning of decoding of each P, SP, or B slice.

在开始解码 P、SP 或 B slice 之前，调用本处理过程。

Outputs of this process are a reference picture list RefPicList0 and, when decoding a B slice, a second reference picture list RefPicList1.

本过程的输出是一参考图像列表 RefPicList0，当解码 B slice 时，还有另外一个参考图像列表 RefPicList1

Decoded reference pictures are marked as "used for short-term reference" or "used for long-term reference" as specified by the bitstream and specified in subclause 8.2.5. Short-term decoded reference pictures are identified by the value of frame_num. Long-term decoded reference pictures are assigned a long-term frame index as specified by the bitstream and specified in subclause 8.2.5.

根据码流，解码参考图像被标识为“用作短期参考”或者“用作长期参考”，如 8.2.5 小节所示。短期解码参考图像用 frame_num 的值来标识，长期解码参考图像则根据码流分配给一个长期帧索引，如 8.2.5 小节所示。

Subclause 8.2.4.1 specifies

- the assignment of variables FrameNum and FrameNumWrap to each of the short-term reference frames,
- the assignment of variable PicNum to each of the short-term reference pictures, and
- the assignment of variable LongTermPicNum to each of the long-term reference pictures.

8.2.4.1 小节主要描述了：

- 对每一个短期参考帧的 FrameNum 和 FrameNumWrap 变量的赋值过程；
- 对每一个短期参考帧的 PicNum 变量的赋值过程；
- 对每一个长期参考图像的 LongTermPicNum 变量的赋值过程；

Reference pictures are addressed through reference indices as specified in subclause 8.4.2.1. A reference index is an index into a list of variables PicNum and LongTermPicNum, which is called a reference picture list. When decoding a P or SP slice, there is a single reference picture list RefPicList0. When decoding a B slice, there is a second independent reference picture list RefPicList1 in addition to RefPicList0.

参考图像由 8.4.2.1 小节所说的参考索引来访问，一个参考索引是一个变量 PicNum或者 LongTermPicNum 列表的索引，他们通常被称为参考图像列表。当解码 P或 SP slice时，只有一个参考图像列表 RefPicList0，当解码 B slice 时，除了 RefPicList0 之外还有另一个独立的参考图像列表 RefPicList1 。

Let LongTermEntry(RefPicListX[i]) for an entry RefPicListX[i] at index i in reference picture list X where X is 0 or 1 be specified as equal to 1 if RefPicListX[i] is associated with a LongTermPicNum (for a long-term reference picture) and be specified as equal to 0 if the entry is associated with a PicNum (for a short-term reference picture).

对于参考图像列表 X 索引为 i 的 RefPicListX[i]来说， X 的范围是 0 或 1， 如果RefPicListX[i] 和 LongTermPicNum 相关（对于长期参考图像），那么 LongTermEntry(RefPicListX[i]) 的值被设置为 1 ，如果 RefPicListX[i] 和 PicNum 相关（对于短期参考图像），那么LongTermEntry(RefPicListX[i]) 的值被设置为 0 。

At the beginning of decoding of each slice, reference picture list RefPicList0, and for B slices RefPicList1, are derived as follows.

- An initial reference picture list RefPicList0 and for B slices RefPicList1 are derived as specified in subclause 8.2.4.2.
- The initial reference picture list RefPicList0 and for B slices RefPicList1 are modified as specified in subclause 8.2.4.3.

在开始解码每一个 slice 时，参考图像列表 RefPicList0，对于 B slice，还有 RefPicList1 的计算如下：

- 初始化参考图像列表 RefPicList0，对于 B slice 还有 RefPicList1 ，如 8.2.4.2 小节所示；
- 修改初始化的参考图像列表 RefPicList0，对于 B slice 还有 RefPicList1 ，如 8.2.4.3 小节所示。

The number of entries in the modified reference picture list RefPicList0 is num_ref_idx_l0_active_minus1 + 1, and for B slices the number of entries in the modified reference picture list RefPicList1 is num_ref_idx_l1_active_minus1 + 1. A reference picture may appear at more than one index in the modified reference picture lists RefPicList0 or RefPicList1.

在修改的参考图像列表 RefPicList0 中，索引的数目是 num_ref_idx_l0_active_minus1 + 1，对于 B slice，在修改的参考图像列表 RefPicList1 中，索引的数目是 num_ref_idx_l1_active_minus1 + 1。在修改的参考图像索引列表 RefPicList0 或 RefPicList1中，一个参考图像可以对应多个索引。

8.2.4.1 Decoding process for picture numbers

8.2.4.1 图像数的解码过程

The variables FrameNum, FrameNumWrap, PicNum, LongTermFrameIdx, and LongTermPicNum are used for the initialisation process for reference picture lists in subclause 8.2.4.2, the modification process for reference picture lists in subclause 8.2.4.3, and for the decoded reference picture marking process in subclause 8.2.5.

参考图像列表变量 FrameNum, FrameNumWrap, PicNum, LongTermFrameIdx, and LongTermPicNum 初始化过程的描述见 8.2.4.2 小节，修改过程的描述见 8.2.4.3 小节，而对解码图像的标识过程如 8.2.5 所示。

To each short-term reference picture the variables FrameNum and FrameNumWrap are assigned as follows. First, FrameNum is set equal to the syntax element frame_num that has been decoded in the slice header(s) of the corresponding short-term reference picture. Then the variable FrameNumWrap is derived as

对于每一个短期参考图像来说，变量 FrameNum 和 FrameNumWrap 的赋值过程如下：首先，FrameNum被设置成对应短期参考图像的 slice 头中的语法元素 frame_num 的值。变量 FrameNumWrap 的计算方法如下：

```
if( FrameNum > frame_num )
    FrameNumWrap = FrameNum – MaxFrameNum                    (8-28)
else
    FrameNumWrap = FrameNum
```

where the value of frame_num used in Equation 8-28 is the frame_num in the slice header(s) for the current picture.

在等式 8-28 中，frame_num 的值为当前图像的 slice 头中的 frame_num。

To each long-term reference picture the variable LongTermFrameIdx is assigned as specified in subclause 8.2.5.

对于每一个长期参考图像来说，变量 LongTermFrameIdx 的值按 8.2.5 小节说明的那样进行赋值。

To each short-term reference picture a variable PicNum is assigned, and to each long-term reference picture a variable LongTermPicNum is assigned. The values of these variables depend on the value of field_pic_flag and bottom_field_flag for the current picture and they are set as follows.

对于每一个短期参考图像来说，变量 PicNum 被赋值，而对于每一个长期参考图像来说，变量 LongTermPicNum 被赋值。这些变量的值与当前图像的 field_pic_flag 和 bottom_field_flag 的值相关，它们的赋值方法如下：

- If field_pic_flag is equal to 0,
- 如果 field_pic_flag 等于 0，
  - For each short-term reference frame or complementary reference field pair:
  - 对于每一个短期参考帧或互补的参考场对：

PicNum = FrameNumWrap (8-29)

  - For each long-term reference frame or long-term complementary reference field pair:
  - 对于每一个长期的参考帧或互补的参考场对：

LongTermPicNum = LongTermFrameIdx (8-30)

NOTE – When decoding a frame the value of MbaffFrameFlag has no influence on the derivations in subclauses 8.2.4.2, 8.2.4.3, and 8.2.5.
注－ 当解码一帧图像时，MbaffFrameFlag 的值对 8.2.4.2, 8.2.4.3, 和 8.2.5小节中的计算没有影响。

- Otherwise (field_pic_flag is equal to 1),
- 否则（field_pic_flag 等于 1）（注意：这里最好用图示表示）
  - For each short-term reference field the following applies
  - 对于每一个短期参考图像，如下所示：
    - If the reference field has the same parity as the current field
    - 如果参考场和当前场是相同的（同是底场或顶场）

PicNum = 2 * FrameNumWrap + 1 (8-31)

    - Otherwise (the reference field has the opposite parity of the current field)
    - 否则（参考场和当前场是相异的，如一个为顶场，一个为底场的情况）

PicNum = 2 * FrameNumWrap (8-32)

  - For each long-term reference field the following applies
  - 对于每一个长期参考图像，如下所示
    - If the reference field has the same parity as the current field
    - 如果参考场和当前场是相同的（同是底场或顶场）

LongTermPicNum = 2 * LongTermFrameIdx + 1 (8-33)

    - Otherwise (the reference field has the opposite parity of the current field)
    - 否则（参考场和当前场是相异的，如一个为顶场，一个为底场的情况）

LongTermPicNum = 2 * LongTermFrameIdx (8-34)

8.2.4.2 Initialisation process for reference picture lists

8.2.4.2 参考图像列表的初始化过程

This initialisation process is invoked when decoding a P, SP, or B slice header.

当解码一 P、SP 或 B slice 头时，使用这个初始化过程。

Outputs of this process are initial reference picture list RefPicList0, and when decoding a B slice, initial reference picture list RefPicList1.

这个过程的输出是初始化的图像参考列表 RefPicList0，当解码 B slice时，还有初始化的参考图像列表 RefPicList1

RefPicList0 and RefPicList1 have initial entries of the variables PicNum and LongTermPicNum as specified in subclauses 8.2.4.2.1 through 8.2.4.2.5.

如 8.2.4.2.1 到 8.2.4.2.5 所描述的那样，RefPicList0 和 RefPicList1有变量 PicNum 和 LongTermPicNum的初始值。

When the number of entries in the initial RefPicList0 or RefPicList1 produced as specified in subclauses 8.2.4.2.1 through 8.2.4.2.5 is greater than num_ref_idx_l0_active_minus1 + 1 or num_ref_idx_l1_active_minus1 + 1, respectively, the extra entries past position num_ref_idx_l0_active_minus1 or num_ref_idx_l1_active_minus1 are discarded from the initial reference picture list.

当由 8.2.4.2.1 到 8.2.4.2.5 小节所产生的 RefPicList0 或 RefPicList1 中的索引数目大于 num_ref_idx_l0_active_minus1 + 1 或者 num_ref_idx_l1_active_minus1 + 1 时，那些大于 num_ref_idx_l0_active_minus1 + 1 或者 num_ref_idx_l1_active_minus1 + 1位置处的索引将被从初始化的参考图像列表中去除。

When the number of entries in the initial RefPicList0 or RefPicList1 produced as specified in subclauses 8.2.4.2.1 through 8.2.4.2.5 is less than num_ref_idx_l0_active_minus1 + 1 or num_ref_idx_l1_active_minus1 + 1, respectively, the remaining entries in the initial reference picture list are set equal to "no reference picture".

当由 8.2.4.2.1 到 8.2.4.2.5 小节所产生的 RefPicList0 或 RefPicList1 中的索引数目小于 num_ref_idx_l0_active_minus1 + 1 或者 num_ref_idx_l1_active_minus1 + 1时，初始参考图像列表中的剩余位置的索引被设置为“没有参考图像（no reference picture）”

8.2.4.2.1 Initialisation process for the reference picture list for P and SP slices in frames

8.2.4.2.1 对按帧解码的 P 和 SP slice，参考图像列表的初始化过程

This initialisation process is invoked when decoding a P or SP slice in a coded frame.

当解码一帧结构的 P或 SP slice 时，使用这个初始化过程。

Output of this process is the initial reference picture list RefPicList0.

这个过程的输出是初始化参考图像列表 RefPicList0。

The reference picture list RefPicList0 is ordered so that short-term reference frames and short-term complementary reference field pairs have lower indices than long-term reference frames and long-term complementary reference field pairs.

参考图像列表 RefPicList0被排序，使短期参考帧和短期互补参考场对比长期参考帧和长期互补参考场对有较小的索引值。

The short-term reference frames and complementary reference field pairs are ordered starting with the frame or complementary field pair with the highest PicNum value and proceeding through in descending order to the frame or complementary field pair with the lowest PicNum value.

短期参考帧和互补的参考场对按降序从最大 PicNum 的帧或互补场对进行排列，一直到最小 PicNum 的帧或互补场对。

The long-term reference frames and complementary reference field pairs are ordered starting with the frame or complementary field pair with the lowest LongTermPicNum value and proceeding through in ascending order to the frame or complementary field pair with the highest LongTermPicNum value.

长期参考帧和互补参考场对按升序从最小 LongTermPicNum 的帧或互补场对进行排列，一直到最大 LongTermPicNum的帧或互补场对。

NOTE – A non-paired reference field is not used for inter prediction for decoding a frame, regardless of the value of MbaffFrameFlag.
注 — 在帧结构的解码过程中，不管 MbaffFrameFlag的值是多少，帧间预测都不使用不成对的场做参考。

For example, when three reference frames are marked as "used for short-term reference" with PicNum equal to 300, 302, and 303 and two reference frames are marked as "used for long-term reference" with LongTermPicNum equal to 0 and 3, the initial index order is:

- RefPicList0[0] is set equal to PicNum = 303,
- RefPicList0[1] is set equal to PicNum = 302,
- RefPicList0[2] is set equal to PicNum = 300,
- RefPicList0[3] is set equal to LongTermPicNum = 0, and

- RefPicList0[4] is set equal to LongTermPicNum = 3.

And LongTermEntry(RefPicList0[i]) is set equal to 0 for i equal to 0, 1, and 2; and is set equal to 1 for i equal to 3 and 4.

例如，当三个参考帧被标识为“用作短期参考”，它们的 PicNum 的值分别为 300、302 和 303，两个参考图像被标识为“用作长期参考”，它们的 LongTermPicNum 分别为 0 和 3，那么对应的索引顺序应该是：

- RefPicList0[0] 被设置成 PicNum = 303,
- RefPicList0[1] 被设置成 PicNum = 302,
- RefPicList0[2] 被设置成 PicNum = 300,
- RefPicList0[3] 被设置成 LongTermPicNum = 0, 并且
- RefPicList0[4] 被设置成 LongTermPicNum = 3.

并且当 i 等于 0、1 和 2 时，LongTermEntry(RefPicList0[i]) 的值被设置成 0，而当 i 等于 3 和 4 时，它的值则被设置成 1。

8.2.4.2.2 Initialisation process for the reference picture list for P and SP slices in fields

8.2.4.2.2 对按场解码的 P 和 SP slice，参考图像列表的初始化过程

This initialisation process is invoked when decoding a P or SP slice in a coded field.

当解码一场结构的 P或 SP slice 时，使用这个初始化过程。

Output of this process is reference picture list RefPicList0.

这个过程的输出是初始化参考图像列表 RefPicList0。

When decoding a field, each field included in the reference picture list has a separate index in the list.

当解码一场图像时，包含在参考图像列表中的每一场都有一单一索引。

NOTE - When decoding a field, there are effectively at least twice as many pictures available for referencing as there would be when decoding a frame at the same position in decoding order.

注 一 当解码场结构图像时，和按相同的解码顺序在相同的位置解码帧结构图像时相比，至少有两倍的图像可用作参考图像。

Two ordered lists of reference frames, refFrameList0ShortTerm and refFrameList0LongTerm, are derived as follows. For purposes of the formation of this list of frames, decoded frames, complementary reference field pairs, non-paired reference fields and reference frames in which a single field is marked "used for short-term reference" or "used for long-term reference" are all considered reference frames.

参考图像的两个列表 refFrameList0ShortTerm 和 refFrameList0LongTerm按下面的方法进行排列。为了构造帧，解码帧和互补参考场对列表的目的，那些只有一个场（底场或顶场）标识为“用作短期参考”或“用作长期参考”的参考帧或非成对的参考场都被认为是参考帧。

- The FrameNumWrap of all frames having one or more field marked "used for short-term reference" are included in the list of short-term reference frames refFrameList0ShortTerm. When the current field is the second field (in decoding order) of a complementary reference field pair and the first field is marked as "used for short-term reference", the FrameNumWrap of the current field is included in the list refFrameList0ShortTerm. refFrameList0ShortTerm is ordered starting with the frame with the highest FrameNumWrap value and proceeding through in descending order to the frame with the lowest FrameNumWrap value.

- 有一个或多个场被标识为“用作短期参考”的所有帧的 FrameNumWrap都被包括在短期参考图像列表 refFrameList0ShortTerm 中，当当前场是一互补参考场对的第二场（按解码顺序）并且第一场被标识为“用作短期参考”，那么当前场的 FrameNumWrap 也被包含在列表 refFrameList0ShortTerm 中。refFrameList0ShortTerm 按降序从最大的 FrameNumWrap 的帧开始排序，一直到最小的 FrameNumWrap 的帧。

- The LongTermFrameIdx of all frames having one or more field marked "used for long-term reference" are included in the list of long-term reference frames refFrameList0LongTerm. When the current field is the second field (in decoding order) of a complementary reference field pair and the first field is marked as "used for long-term reference", the LongTermFrameIdx of the first field is included in the list refFrameList0LongTerm. refFrameList0LongTerm is ordered starting with the frame with the lowest LongTermFrameIdx value and proceeding through in ascending order to the frame with the highest LongTermFrameIdx value.

- 有一个或多个场被标识为“用作长期参考”的所有帧的 LongTermFrameIdx 都被包括在长期参考图像列表 refFrameList0LongTerm 中，当当前场是一互补参考场对的第二场（按解码顺序）并且第一场被标识为“用作短期参考”，那么第一场的 LongTermFrameIdx 也被包括在列表 refFrameList0LongTerm 中。refFrameList0LongTerm 按升序从最小的 LongTermFrameIdx的帧开始排序，一直到最大的 LongTermFrameIdx 的帧。

The process specified in subclause 8.2.4.2.5 is invoked with refFrameList0ShortTerm and refFrameList0LongTerm given as input and the output is assigned to RefPicList0.

将 refFrameList0ShortTerm 和 refFrameList0LongTerm 作为输入，使用 8.2.4.2.5 小节的方法，输出赋给 RefPicList0。

8.2.4.2.3 Initialisation process for reference picture lists for B slices in frames

8.2.4.2.3 对帧结构解码的 B slice，参考图像列表的初始化过程

This initialisation process is invoked when decoding a B slice in a coded frame.

当解码一帧结构的 B slice 时，使用这个初始化过程。

Outputs of this process are the initial reference picture lists RefPicList0 and RefPicList1.

这个过程的输出是初始化参考图像列表 RefPicList0 和 RefPicList1。

For B slices, the order of short-term reference pictures in the reference picture lists RefPicList0 and RefPicList1 depends on output order, as given by PicOrderCnt( ).

对于 B slice，在参考图像列表 RefPicList0 和 RefPicList1中，短期参考图像的顺序和输出顺序有关，由函数 PicOrderCnt( )计算得到。

The reference picture list RefPicList0 is ordered such that short-term reference frames and short-term complementary reference field pairs have lower indices than long-term reference frames and long-term complementary reference field pairs. It is derived as follows.

排序参考图像列表 RefPicList0，使短期参考帧和短期互补参考场对比长期参考帧和长期互补参考场对有更小的索引，具体的方法如下：

- Short-term reference frames and short-term complementary reference field pairs are ordered starting with the short-term reference frame or complementary reference field pair frm0 with the largest value of PicOrderCnt(frm0) less than the value of PicOrderCnt(CurrPic) and proceeding through in descending order to the short-term reference frame or complementary reference field pair frm1 that has the smallest value of PicOrderCnt(frm1), and then continuing with the short-term reference frame or complementary reference field pair frm2 with the smallest value of PicOrderCnt(frm2) greater than the value of PicOrderCnt(CurrPic) of the current frame and proceeding through in ascending order to the short-term reference frame or complementary reference field pair frm3 that has the largest value of PicOrderCnt(frm3).

- 短期参考帧和短期互补参考场对按降序先排列PicOrderCnt( )比PicOrderCnt(CurrPic)小的短期参考帧或互补参考场对，从PicOrderCnt(frm0)最大的短期参考帧或互补参考场对开始，一直到PicOrderCnt(frm1)最小的短期参考帧或互补参考场对；然后再按升序排列PicOrderCnt(frm0)比PicOrderCnt(CurrPic)大的短期参考帧或互补参考场对，从PicOrderCnt(frm2)最小的短期参考帧或互补参考场对开始，一直到PicOrderCnt(frm3)最大的短期参考帧或互补参考场对；

- The long-term reference frames and long-term complementary reference field pairs are ordered starting with the long-term reference frame or complementary reference field pair that has the lowest LongTermPicNum value and proceeding through in ascending order to the long-term reference frame or complementary reference field pair that has the highest LongTermPicNum value.

- 长期参考帧和长期互补参考场对按升序进行排列，从 LongTermPicNum 最小的长期参考帧或互补参考场对开始，一直到 LongTermPicNum最大的长期参考帧或互补参考场对；

The reference picture list RefPicList1 is ordered so that short-term reference frames and short-term complementary reference field pairs have lower indices than long-term reference frames and long-term complementary reference field pairs. It is derived as follows.

排序参考图像列表 RefPicList1，使短期参考帧和短期互补参考场对比长期参考帧和长期互补参考场对有更小的索引，具体的方法如下：

- Short-term reference frames and short-term complementary reference field pairs are ordered starting with the short-term reference frame or complementary reference field pair frm4 with the smallest value of PicOrderCnt(frm4) greater than the value of PicOrderCnt(CurrPic) of the current frame and proceeding through in ascending order to the short-term reference frame or complementary reference field pair frm5 that has the largest value of PicOrderCnt(frm5), and then continuing with the short-term reference frame or complementary reference field pair frm6 with the largest value of PicOrderCnt(frm6) less than the value of PicOrderCnt(CurrPic) of the current frame and proceeding through in descending order to the short-term reference frame or complementary reference field pair frm7 that has the smallest value of PicOrderCnt(frm7).

- 短期参考帧和短期互补参考场对按降序先排列PicOrderCnt( )比PicOrderCnt(CurrPic)小的短期参考帧或互补参考场对，从PicOrderCnt(frm4)最大的短期参考帧或互补参考场对开始，一直到PicOrderCnt(frm5)最小的短期参考帧或互补参考场对；然后再按升序排列PicOrderCnt(frm0)比PicOrderCnt(CurrPic)大的短期参考帧或互补参考场对，从PicOrderCnt(frm6)最小的短期参考帧或互补参考场对开始，一直到PicOrderCnt(frm7)最大的短期参考帧或互补参考场对；

- Long-term reference frames and long-term complementary reference field pairs are ordered starting with the long-term reference frame or complementary reference field pair that has the lowest LongTermPicNum value and proceeding through in ascending order to the long-term reference frame or complementary reference field pair that has the highest LongTermPicNum value.

- 长期参考帧和长期互补参考场对按升序进行排列，从 LongTermPicNum 最小的长期参考帧或互补参考场对开始，一直到 LongTermPicNum最大的长期参考帧或互补参考场对；

- When the reference picture list RefPicList1 has more than one entry and it is identical to the reference picture list RefPicList0, the first two entries RefPicList1[0] and RefPicList1[1] are switched.

- 当参考图像列表 RefPicList1有多个索引（大于一个）并且和参考图像列表 RefPicList0 相同，那么最初的两个索引 RefPicList1[0] 和 RefPicList1[1]需要互换。

  NOTE – A non-paired reference field is not used for inter prediction of frames independent of the value of MbaffFrameFlag.

  注 － 不管 MbaffFrameFlag 的值如何，在帧结构的帧间预测中都不使用非成对的参考场做预测。

8.2.4.2.4 Initialisation process for reference picture lists for B slices in fields

8.2.4.2.4 对场结构解码的 B slice，参考图像列表的初始化过程

This initialisation process is invoked when decoding a B slice in a coded field.

当解码一场结构的 B slice 时，使用这个初始化过程。

Outputs of this process are the initial reference picture lists RefPicList0 and RefPicList1.

这个过程的输出是初始化参考图像列表 RefPicList0 和 RefPicList1。

When decoding a field, each field of a stored reference frame is identified as a separate reference picture with a unique index. The order of short-term reference pictures in the reference picture lists RefPicList0 and RefPicList1 depend on output order, as given by PicOrderCnt( ).

当解码一场图像时，被保存的参考帧的每一场（底场和顶场）都作为一独立的参考图像，并且具有唯一的索引，在参考图像列表 RefPicList0 和 RefPicList1中，短期参考图像的顺序和输出顺序有关，由函数 PicOrderCnt( ) 计算得到。

  NOTE – When decoding a field, there are effectively at least twice as many pictures available for referencing as there would be when decoding a frame at the same position in decoding order.

  注 － 当解码场结构图像时，和按相同的解码顺序在相同的位置解码帧结构图像时相比，至少有两倍的图像可用作参考图像。

Three ordered lists of reference frames, refFrameList0ShortTerm, refFrameList1ShortTerm and refFrameListLongTerm, are derived as follows. For purposes of the formation of these lists of frames the term reference entry refers in the following to decoded reference frames, complementary reference field pairs, or non-paired reference fields.

下面将导出三个参考帧的排序列表 refFrameList0ShortTerm, refFrameList1ShortTerm 和 refFrameListLongTerm，为了构造帧的列表，那些互补的参考场对或者非成对的参考场都被称为解码参考场。

- refFrameList0ShortTerm is ordered starting with the reference entry f0 with the largest value of PicOrderCnt(f0) less than or equal to the value of PicOrderCnt(CurrPic) of the current field and proceeding through in descending order to the short-term reference entry f1 that has the smallest value of PicOrderCnt(f1), and then continuing with the reference entry f2 with the smallest value of PicOrderCnt(f2) greater than the value of PicOrderCnt(CurrPic) of the current field and proceeding through in ascending order to the short-term reference entry f3 that has the largest value of PicOrderCnt(f3).

- refFrameList0ShortTerm 先按降序对 PicOrderCnt（）小于当前场的PicOrderCnt(CurrPic)的短期参考场进行排列，从PicOrderCnt(f0)最大的短期参考场开始，一直到PicOrderCnt(f1)最小的短期参考场；然后再按升序对 PicOrderCnt（）大于当前场的PicOrderCnt(CurrPic)的短期参考场进行排列，从PicOrderCnt(f2)最小的短期参考场开始，一直到PicOrderCnt(f3)最大的短期参考场。

  NOTE - When for the current field nal_ref_idc is greater than 0 and the current coded field follows in decoding order a coded field fld1 with which together it forms a complementary reference field pair after decoding, fld1 shall be included into the list refFrameList0ShortTerm using PicOrderCnt(fld1) and the ordering method described in the previous sentence shall be applied.

  注 一 当当前场的 nal_ref_idc 大于 0 并且按照解码顺序当前编码场还是一互补参考场对的第二场，那么该互补参考场对的第一场 fld1 应该包括在列表 refFrameList0ShortTerm中，顺序有PicOrderCnt(fld1)确定，然后在使用上述的排序方法。

- refFrameList1ShortTerm is ordered starting with the reference entry f4 with the smallest value of PicOrderCnt(f4) greater than the value of PicOrderCnt(CurrPic) of the current field and proceeding through in ascending order to the short-term reference entry f5 that has the largest value of PicOrderCnt(f5), and then continuing with the reference entry f6 with the largest value of PicOrderCnt(f6) less than or equal to the value of PicOrderCnt(CurrPic) of the current field and proceeding through in descending order to the short-term reference entry f7 that has the smallest value of PicOrderCnt(f7).

- refFrameList1ShortTerm 先按降序对 PicOrderCnt（）小于当前场的PicOrderCnt(CurrPic)的短期参考场进行排列，从PicOrderCnt(f4)最大的短期参考场开始，一直到PicOrderCnt(f5)最小的短期参考场；然后再按升序对 PicOrderCnt（）大于当前场的PicOrderCnt(CurrPic)的短期参考场进行排列，从PicOrderCnt(f6)最小的短期参考场开始，一直到PicOrderCnt(f7)最大的短期参考场。

  NOTE - When for the current field nal_ref_idc is greater than 0 and the current coded field follows in decoding order a coded field fld2 with which together it forms a complementary reference field pair after decoding, fld2 shall be included into the list refFrameList1ShortTerm using PicOrderCnt(fld2) and the ordering method described in the previous sentence shall be applied.

  注 一 当当前场的 nal_ref_idc 大于 0 并且按照解码顺序当前编码场还是一互补参考场对的第二场，那么该互补参考场对的第一场 fld1 应该包括在列表 refFrameList0ShortTerm中，顺序有PicOrderCnt(fld1)确定，然后在使用上述的排序方法。

- refFrameListLongTerm is ordered starting with the reference entry having the lowest LongTermFrameIdx value and proceeding through in ascending order to the reference entry having highest LongTermPicNum value.

- refFrameListLongTerm 按升序进行排列，从最小的 LongTermFrameIdx 长期参考场开始，一直到最大的 LongTermFrameIdx 长期参考场

  NOTE - When the complementary field of the current picture is marked "used for long-term reference" it is included into the list refFrameListLongTerm. A reference entry in which only one field is marked as “used for long-term reference” is included into the list refFrameListLongTerm.

  注 一 当当前图像的互补场被标识为“用作长期参考”，那么它也被包括在 refFrameListLongTerm 中，只有一场被标识为“用作长期参考”（另外一场没有标识）的图像也被包括在列表 refFrameListLongTerm中。

The process specified in subclause 8.2.4.2.5 is invoked with refFrameList0ShortTerm and refFrameListLongTerm given as input and the output is assigned to RefPicList0.

将 refFrameList0ShortTerm 和 refFrameListLongTerm作为 8.2.4.2.5 小节的输入，输出再赋给 RefPicList0。

The process specified in subclause 8.2.4.2.5 is invoked with refFrameList1ShortTerm and refFrameListLongTerm given as input and the output is assigned to RefPicList1.

将 refFrameList0ShortTerm 和 refFrameListLongTerm作为 8.2.4.2.5 小节的输入，输出再赋给 RefPicList1。

When the reference picture list RefPicList1 has more than one entry and it is identical to the reference picture list RefPicList0, the first two entries RefPicList1[0] and RefPicList1[1] are switched.

当参考图像列表 RefPicList1有多个索引（大于一个）并且和参考图像列表 RefPicList0 相同，那么最初的两个索引 RefPicList1[0] 和 RefPicList1[1]需要互换。

8.2.4.2.5 Initialisation process for reference picture lists in fields

8.2.4.2.5 对按场解码的参考图像列表的初始化过程

Inputs of this process are the reference frame lists refFrameListXShortTerm (with X may be 0 or 1) and refFrameListLongTerm.

本过程的输入是参考帧列表 refFrameListXShortTerm（X 为 0 或者 1）和 refFrameListLongTerm。

Output of this process is reference picture list RefPicListX (which may be RefPicList0 or RefPicList1).

本过程的输出是参考图像列表 RefPicListX（RefPicList0 或者RefPicList1）。

The reference picture list RefPicListX is a list ordered such that short-term reference fields have lower indices than long-term reference fields. Given the reference frame lists refFrameListXShortTerm and refFrameListLongTerm, it is derived as follows.

排列参考图像列表，使短期参考场的索引比长期参考场的索引要小。根据给定的参考帧列表 refFrameListXShortTerm 和 refFrameListLongTerm，参考图像列表如下：（最好用图示说明）

- Short-term reference fields are ordered by selecting reference fields from the ordered list of frames refFrameListXShortTerm by alternating between fields of differing parity, starting with fields that have the same parity as the current field. When one field of a reference frame was not decoded or is not marked as "used for short-term reference", the missing field is ignored and instead the next available stored reference field of the chosen parity from the ordered list of frames refFrameListXShortTerm is inserted into RefPicListX. When there are no more short-term reference fields of the alternate parity in the ordered list of frames refFrameListXShortTerm, the next not yet indexed fields of the available parity are inserted into RefPicListX in the order in which they occur in the ordered list of frames refFrameListXShortTerm.

- 从已经排序的帧列表 refFrameListXShortTerm 中选择参考场对短期参考场进行排序，从与当前场相对应的场开始（如：当前场为顶场，那么就从顶场开始排序），互补的场（顶场和底场）交叉排列。当某一帧的一个场没有解码或者没有被标识为"用作短期参考"，那么就忽略该场，用帧列表 refFrameListXShortTerm 中下一个可用的并且和忽略的场相对应的场来替代（如：如果忽略的场是底场，那么替代的应该也是底场）。当帧列表 refFrameListXShortTerm中没有互补的场（如：剩余的都是顶场），那么对这些还没有排列的场按它们在帧列表 refFrameListXShortTerm中排列的顺序进行排列。

- Long-term reference fields are ordered by selecting reference fields from the ordered list of frames refFrameListLongTerm by alternating between fields of differing parity, starting with fields that have the same parity as the current field. When one field of a reference frame was not decoded or is not marked as "used for long-term reference", the missing field is ignored and instead the next available stored reference field of the chosen parity from the ordered list of frames refFrameListLongTerm is inserted into RefPicListX. When there are no more long-term reference fields of the alternate parity in the ordered list of frames refFrameListLongTerm, the next not yet indexed fields of the available parity are inserted into RefPicListX in the order in which they occur in the ordered list of frames refFrameListLongTerm.

- 从已经排序的帧列表 refFrameListLongTerm 中选择参考场来对长期参考场进行排序，从与当前场相对应的场开始（如：当前场为顶场，那么就从顶场开始排序），互补的场（顶场和底场）交叉排列。当某一帧的一个场没有解码或者没有被标识为"用作短期参考"，那么就忽略该场，用帧列表 refFrameListLongTerm中下一个可用的并且和忽略的场相对应的场来替代（如：如果忽略的场是底场，那么替代的应该也是底场）。当帧列表 refFrameListLongTerm中没有互补的场（如：剩余的都是顶场），那么对这些还没有排列的场按它们在帧列表 refFrameListLongTerm中排列的顺序进行排列。

8.2.4.3 Reordering process for reference picture lists

8.2.4.3 参考图像列表的重排序过程

Input to this process is reference picture list RefPicList0 and, when decoding a B slice, also reference picture list RefPicList1.

这个过程的输入是参考图像列表 RefPicList0，当解码 B slice时，还有参考图像列表 RefPicList1。

Outputs of this process are a possibly modified reference picture list RefPicList0 and, when decoding a B slice, also a possibly modified reference picture list RefPicList1.

这个过程的输出是可能修改的参考图像列表 RefPicList0，当解码 B slice时，还有可能修改的图像列表 RefPicList1。

When ref_pic_list_reordering_flag_l0 is equal to 1, the following applies.

- Let refIdxL0 be an index into the reference picture list RefPicList0. It is initially set equal to 0.

- The corresponding syntax elements reordering_of_pic_nums_idc are processed in the order they occur in the bitstream. For each of these syntax elements, the following applies.

- If reordering_of_pic_nums_idc is equal to 0 or equal to 1, the process specified in subclause 8.2.4.3.1 is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.

- If reordering_of_pic_nums_idc is equal to 2, the process specified in subclause 8.2.4.3.2 is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.

- Otherwise (reordering_of_pic_nums_idc is equal to 3), the reordering process for reference picture list RefPicList0 is finished.

当 ref_pic_list_reordering_flag_l0 等于 1 时，则如下处理：

- 假设 refIdxL0 是参考图像列表 RefPicList0 的索引，并且被初始化成 0；
- 按照 reordering_of_pic_nums_idc 在码流中出现的顺序依次进行处理。对于每一个语法元素 reordering_of_pic_nums_idc，处理如下：
  - 如果 reordering_of_pic_nums_idc等于 0 或 1，那么使用 8.2.4.3.1小节中的处理过程，将RefPicList0 和 refIdxL0作为输入，输出赋给 RefPicList0 和 refIdxL0；
  - 如果 reordering_of_pic_nums_idc等于 2，那么使用 8.2.4.3.2小节中的处理过程，将RefPicList0 和 refIdxL0作为输入，输出赋给 RefPicList0 和 refIdxL0；
  - 否则（reordering_of_pic_nums_idc等于 3），就结束参考图像列表的重排序过程。

When ref_pic_list_reordering_flag_l1 is equal to 1, the following applies.

- Let refIdxL1 be an index into the reference picture list RefPicList1. It is initially set equal to 0.
- The corresponding syntax elements reordering_of_pic_nums_idc are processed in the order they occur in the bitstream. For each of these syntax elements, the following applies.
  - If reordering_of_pic_nums_idc is equal to 0 or equal to 1, the process specified in subclause 8.2.4.3.1 is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.
  - If reordering_of_pic_nums_idc is equal to 2, the process specified in subclause 8.2.4.3.2 is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.
  - Otherwise (reordering_of_pic_nums_idc is equal to 3), the reordering process for reference picture list RefPicList1 is finished.

当 ref_pic_list_reordering_flag_l1 等于 1 时，则如下处理:

- 假设 refIdxL1 是参考图像列表 RefPicList1 的索引，并且被初始化成 0；
- 按照 reordering_of_pic_nums_idc 在码流中出现的顺序依次进行处理。对于每一个语法元素 reordering_of_pic_nums_idc，处理如下：
  - 如果 reordering_of_pic_nums_idc等于 0 或 1，那么使用 8.2.4.3.1小节中的处理过程，将RefPicList1 和 refIdxL1作为输入，输出赋给 RefPicList1 和 refIdxL1；
  - 如果 reordering_of_pic_nums_idc等于 2，那么使用 8.2.4.3.2小节中的处理过程，将RefPicList1 和 refIdxL1作为输入，输出赋给 RefPicList1 和 refIdxL1；
  - 否则（reordering_of_pic_nums_idc等于 3），就结束参考图像列表的重排序过程。

8.2.4.3.1 Reordering process of reference picture lists for short-term pictures

8.2.4.3.1 短期参考图像列表的重排序过程

Inputs to this process are reference picture list RefPicListX (with X being 0 or 1) and an index refIdxLX into this list.

本过程的输入是参考图像列表RefPicListX(X为0或1)和指向这个列表的索引refIdxLX。

Outputs of this process are a possibly modified reference picture list RefPicListX (with X being 0 or 1) and the incremented index refIdxLX.

本过程的输出是可能被修改的参考图像列表RefPicListX(X为0或1)和增加的索引refIdxLX。

The variable picNumLXNoWrap is derived as follows

变量 picNumLXNoWrap 的计算如下：

- If reordering_of_pic_nums_idc is equal to 0
- 如果 reordering_of_pic_nums_idc 等于 0

```
if( picNumLXPred – ( abs_diff_pic_num_minus1 + 1 ) < 0 )
    picNumLXNoWrap = picNumLXPred – ( abs_diff_pic_num_minus1 + 1 ) + MaxPicNum        (8-35)
else
    picNumLXNoWrap = picNumLXPred – ( abs_diff_pic_num_minus1 + 1 )
```

- Otherwise (reordering_of_pic_nums_idc is equal to 1)
- 否则（reordering_of_pic_nums_idc等于1）

```
if( picNumLXPred + ( abs_diff_pic_num_minus1 + 1 ) >= MaxPicNum )
    picNumLXNoWrap = picNumLXPred + ( abs_diff_pic_num_minus1 + 1 ) – MaxPicNum        (8-36)
else
    picNumLXNoWrap = picNumLXPred + ( abs_diff_pic_num_minus1 + 1 )
```

picNumLXPred is the prediction value for the variable picNumLXNoWrap. When the process specified in this subclause is invoked the first time for a slice (that is, for the first occurrence of reordering_of_pic_nums_idc equal to 0 or 1 in the ref_pic_list_reordering( ) syntax), picNumL0Pred and picNumL1Pred are initially set equal to CurrPicNum. After each assignment of picNumLXNoWrap, the value of picNumLXNoWrap is assigned to picNumLXPred.

PicNumLXPred 是 变量 picNumLXNoWrap 的预测值，对于一个slice来说，如果本小节的处理过程第一次被调用（也就是说，在ref_pic_list_reordering( )语法中第一次出现reordering_of_pic_nums_idc等于0或1的情况），picNumL0Pred 和 picNumL1Pred 被设置成 CurrPicNum，picNumLXNoWrap赋值后，picNumLXNoWrap的值再赋给 picNumLXPred。

The variable picNumLX is derived as follows

变量 picNumLX 的计算如下：

```
if( picNumLXNoWrap > CurrPicNum )
    picNumLX = picNumLXNoWrap – MaxPicNum        (8-37)
else
    picNumLX = picNumLXNoWrap
```

picNumLX shall specify a reference picture that is marked as “used for short-term reference” and shall not specify a short-term reference picture that is marked as "non-existing".

PicNumLX 表示了一标识为“用作短期参考”的参考图像，而不表示标识为“不存在”的短期参考图像。

The following procedure shall then be conducted to place the picture with short-term picture number picNumLX into the index position refIdxLX, shift the position of any other remaining pictures to later in the list, and increment the value of refIdxLX.

下面的程序是将图像数为 picNumLX 的短期图像放到索引位置为 refIdxLX 的参考图像列表 RefPicListX 中，并且将列表中保留的图像统一后移，同时refIdxLX的加一。

```
for( cIdx = num_ref_idx_lX_active_minus1 + 1; cIdx > refIdxLX; cIdx-- )
    RefPicListX[ cIdx ] = RefPicListX[ cIdx – 1]
RefPicListX[ refIdxLX++ ] = picNumLX
nIdx = refIdxLX
for( cIdx = refIdxLX; cIdx <= num_ref_idx_lX_active_minus1 + 1; cIdx++ )        (8-38)
    if( LongTermEntry( RefPicListX[ cIdx ] ) || RefPicListX[ cIdx ] != picNumLX )
        RefPicListX[ nIdx++ ] = RefPicListX[ cIdx ]
```

NOTE – Within this pseudo-code procedure, the length of the list RefPicListX is temporarily made one element longer than the length needed for the final list. After the execution of this procedure, only elements 0 through num_ref_idx_lX_active_minus1 of the list need to be retained.

注 — 在这段伪代码中，列表RefPicListX的长度临时性的大于最终长度一个一个单位，执行完这段程序后，列表中仅仅索引为 0 到 num_ref_idx_lX_active_minus1的元素被保存下来。

8.2.4.3.2 Reordering process of reference picture lists for long-term pictures

8.2.4.3.2 长期参考图像的重排序过程

Inputs to this process are reference picture list RefPicListX (with X being 0 or 1) and an index refIdxLX into this list.

本过程的输入是参考图像列表RefPicListX(X为0或1)和指向这个列表的索引refIdxLX。

Outputs of this process are a possibly modified reference picture list RefPicListX (with X being 0 or 1) and the incremented index refIdxLX.

本过程的输出是可能被修改的参考图像列表RefPicListX(X为0或1)和增加的索引refIdxLX。

LongTermPicNum equal to long_term_pic_num shall specify a reference picture that is marked as "used for long-term reference".

等于 long_term_pic_num 的 LongTermPicNum 表示了一标识“用作长期参考”的参考图像。

The following procedure shall then be conducted to place the picture with long-term picture number long_term_pic_num into the index position refIdxLX, shift the position of any other remaining pictures to later in the list, and increment the value of refIdxLX.

下面的程序是将长期图像数为 long_term_pic_num 的长期图像放到索引位置为 refIdxLX 的参考图像列表 RefPicListX 中，并且将列表中保留的图像统一后移，同时refIdxLX的加一。

```
for( cIdx = num_ref_idx_lX_active_minus1 + 1; cIdx > refIdxLX; cIdx-- )
    RefPicListX[ cIdx ] = RefPicListX[ cIdx – 1]
RefPicListX[ refIdxLX++ ] = LongTermPicNum                                    (8-39)
nIdx = refIdxLX
for( cIdx = refIdxLX; cIdx <= num_ref_idx_lX_active_minus1 + 1; cIdx++ )
    if( !LongTermEntry( RefPicListX[ cIdx ] ) || RefPicListX[ cIdx ] != LongTermPicNum )
        RefPicListX[ nIdx++ ] = RefPicListX[ cIdx ]
```

NOTE – Within this pseudo-code procedure, the length of the list RefPicListX is temporarily made one element longer than the length needed for the final list. After the execution of this procedure, only elements 0 through num_ref_idx_lX_active_minus1 of the list need to be retained.

注 — 在这段伪代码中，列表RefPicListX的长度临时性的大于最终长度一个一个单位，执行完这段程序后，列表中仅仅索引为 0 到 num_ref_idx_lX_active_minus1的元素被保存下来。

8.2.5 Decoded reference picture marking process

8.2.5 解码参考图像的标识过程

This process is invoked for decoded pictures when nal_ref_idc is not equal to 0.

对于 nal_ref_idc 不等于 0 的解码图像使用本过程。

A decoded picture with nal_ref_idc not equal to 0, referred to as a reference picture, is marked as “used for short-term reference” or "used for long-term reference". For a decoded reference frame, both of its fields are marked the same as the frame. For a complementary reference field pair, the pair is marked the same as both of its fields. A picture that is marked as "used for short-term reference" is identified by its FrameNum and, if a field, by its parity. A picture that is marked as "used for long-term reference" is identified by its LongTermFrameIdx and, if a field, by its parity.

nal_ref_idc 不等于 0 的解码图像称为参考图像，被标识为“用作短期参考”或“用作长期参考”。对于一个解码帧，它的两个场（底场和顶场）和帧的标识相同。对于一个互补的参考场对，场对和它的两个场的标识也相同。标识为“用作短期参考”的图像由 FrameNum 索引，如果是场的话，由它对应的部分来索引（每个场都有一个索引），标识为“用作长期参考”的图像由 LongTermFrameIdx 来索引，如果是场的话，也由它的对应部分来索引（每个场都有一个索引）。

Frames or complementary reference field pairs marked as “used for short-term reference” or as "used for long-term reference" can be used as a reference for inter prediction when decoding a frame until the frame or one of its constituent fields is marked as “unused for reference”. A field marked as “used for short-term reference” or as "used for long-term reference" can be used as a reference for inter prediction when decoding a field until marked as “unused for reference”.

在帧结构解码时，标识为“用作短期参考”或“用作长期参考”的帧或互补的参考场对可以用作帧间预测参考，一直到该帧或该互补参考场对中的某一场被标识为“不用作参考场”。在场结构解码时，标识为“用作短期参考”或“用作长期参考”的场可以用作帧间预测，一直到该场被标识为“不用作参考场”。

A picture can be marked as "unused for reference" by the sliding window reference picture marking process, a first-in, first-out mechanism specified in subclause 8.2.5.3 or by the adaptive memory control reference picture marking process, a customised adaptive marking operation specified in subclause 8.2.5.4.

一图像使用下述的方法将其标识为“不用作参考”，如 8.2.5.3 小节采用先进先出机制（FIFO）的“滑动窗口参考图像标识方法”和 8.2.5.4小节采用自定义自适应标识机制的“自适应缓存控制参考图像标识方法”。

A short-term reference picture is identified for use in the decoding process by its picture number PicNum, and a long-term reference picture is identified for use in the decoding process by its long-term picture number LongTermPicNum. Subclause 8.2.4.1 specifies how PicNum and LongTermPicNum are calculated.

在解码过程中，短期参考图像由图像数 PicNum 来索引，而长期参考图像由长期图像数 LongTermPicNum 来索引，8.2.4.1 小节说明了 PicNum 和 LongTermPicNum 的计算方法。

8.2.5.1 Sequence of operations for decoded reference picture marking process

8.2.5.1 解码参考图像标识过程的操作顺序

Decoded reference picture marking proceeds in the following ordered steps.

解码参考图像标识按照如下的步骤进行：

1. When frame_num of the current picture is not equal to PrevRefFrameNum and is not equal to (PrevRefFrameNum + 1) % MaxFrameNum, the decoding process for gaps in frame_num is performed according to subclause 8.2.5.2.
2. All slices of the current picture are decoded.
3. Depending on whether the current picture is an IDR picture, the following applies.
   - If the current picture is an IDR picture, the following applies.
     - All reference pictures shall be marked as "unused for reference"
     - If long_term_reference_flag is equal to 0, the IDR picture shall be marked as "used for short-term reference" and MaxLongTermFrameIdx shall be set equal to “no long-term frame indices”.
     - Otherwise (long_term_reference_flag is equal to 1), the IDR picture shall be marked as "used for long-term reference", the LongTermFrameIdx for the IDR picture shall be set equal to 0, and MaxLongTermFrameIdx shall be set equal to 0.
   - Otherwise (the current picture is not an IDR picture), the following applies.
     - If adaptive_ref_pic_marking_mode_flag is equal to 0, the process specified in subclause 8.2.5.3 is invoked.
     - Otherwise (adaptive_ref_pic_marking_mode_flag is equal to 1), the process specified in subclause 8.2.5.4 is invoked.
4. When the current picture is not an IDR picture and it was not marked as "used for long-term reference" by memory_management_control_operation equal to 6, it is marked as "used for short-term reference".

1. 当当前图像的 frame_num 不等于 PrevRefFrameNum，并且也不等于 (PrevRefFrameNum + 1) % MaxFrameNum 时，就使用 8.2.5.2 小节的关于 frame_num 间隔的解码过程;
2. 解码当前图像的所有 slice;
3. 根据当前图像是不是 IDR 图像，再进行下面的步骤：
   - 如果当前图像是 IDR 图像，使用下面的步骤：
     - 所有的参考图像被标识为“不用作参考”；
     - 如果 long_term_reference_flag 等于 0， 那么该 IDR 图像被标识为“用作短期参考”，并且 MaxLongTermFrameIdx 被设置成“没有长期参考帧”；
     - 否则（long_term_reference_flag 等于 1），该 IDR 图像被标识为“用作长期参考”，它的 LongTermFrameIdx 被设置成 0，MaxLongTermFrameIdx 也被设置成 0；
   - 否则（当前图像不是 IDR 图像），使用下面的步骤：
     - 如果 adaptive_ref_pic_marking_mode_flag 等于 0， 那么就使用 8.2.5.3 小节中处理过程；
     - 否则（adaptive_ref_pic_marking_mode_flag 等于 1），使用 8.2.5.4 小节中的处理过程。
4. 当当前图像不是 IDR 图像，并且也没有因为 memory_management_control_operation 等于 6 而被标识为“用作长期参考”，那么它将被标识为“用作短期参考”。

After marking the current decoded reference picture, the total number of frames with at least one field marked as "used for reference", plus the number of complementary field pairs with at least one field marked as "used for reference", plus the number of non-paired fields marked as "used for reference" shall not be greater than num_ref_frames.

标识完当前解码图像之后，参考帧的总数不应该超过 num_ref_frames ，这里的参考帧包括：至少有一场（底场或顶场）被标识为"用作参考"的帧（frame）、至少有一场（底场或顶场）被标识为"用作参考"的互补场对（complementary field pairs）和标识为"用作参考"的非成对场（non-paired fields）。

8.2.5.2 Decoding process for gaps in frame_num

8.2.5.2 关于 frame_num 间隔的解码过程

This process is invoked when frame_num is not equal to PrevRefFrameNum and is not equal to (PrevRefFrameNum + 1) % MaxFrameNum.

当 frame_num 不等于 PrevRefFrameNum 和 (PrevRefFrameNum + 1) % MaxFrameNum 时，使用本过程。

NOTE – This process can only be invoked for a conforming bitstream if gaps_in_frame_num_value_allowed_flag is equal to 1. If gaps_in_frame_num_value_allowed_flag is equal to 0 and frame_num is not equal to PrevRefFrameNum and is not equal to (PrevRefFrameNum + 1) % MaxFrameNum, the decoding process should infer an unintentional loss of pictures.

注 － 在兼容性的码流中如果 gaps_in_frame_num_value_allowed_flag 等于 1，则使用本过程。如果 gaps_in_frame_num_value_allowed_flag 等于 0，并且 frame_num 不等于 PrevRefFrameNum 和 (PrevRefFrameNum + 1) % MaxFrameNum，那么解码器认为出现了非故意的图像丢失（可以认为是丢包或码流中出现误码）。

When this process is invoked, a set of values of frame_num pertaining to "non-existing" pictures is derived as all values taken on by UnusedShortTermFrameNum in Equation 7-10 except the value of frame_num for the current picture.

当使用本过程时，除了当前图像的 frame_num 的值之外，那些属于"不存在"图像的 frame_num 的值是由等式 7-10 中用 UnusedShortTermFrameNum 计算得到。

The decoding process shall mark a frame for each values of frame_num pertaining to "non-existing" pictures, in the order in which the values of UnusedShortTermFrameNum are generated by Equation 7-10, using the "sliding window" picture marking process as specified in subclause 8.2.5.3. The added frames shall also be marked as "non-existing" and "used for short-term reference". The sample values of the added frames may be set to any value. These added frames which are marked as "non-existing" shall not be referred to in the inter prediction process, shall not be referred to in the reordering commands for reference picture lists for short-term pictures (subclause 8.2.4.3.1), and shall not be referred to in the assignment process of a LongTermFrameIdx to a short-term picture (subclause 8.2.5.4.3). When a frame marked as "non-existing" is marked as "unused for reference" using the "sliding window" buffering process or the "adaptive memory control" mechanism, it shall no longer be marked as "non-existing".

解码过程将标识那些 frame_num属于"不存在"图像的图像帧，标识过程使用 8.2.5.3 小节的"滑动窗口"图像标识方法按照等式 7-10 产生 UnusedShortTermFrameNum 值的顺序进行标识。这些增加的图像被标识为"不存在"并且"用作短期参考"（不能理解），它们的象素值可以被设置成任何值。这些标识为"不存在"的图像不能用作帧间预测过程和 8.2.4.3.1 小节的短期参考图像列表的重排序过程，也不能参与 8.2.5.4.3 小节的给短期图像分配 LongTermFrameIdx 的过程。当将标识为"不存在"的图像使用"滑动窗口"缓存控制或"自适应缓存控制"方法标识为"不用作参考"时，它将不再标识为"不存在"图像。

NOTE - The decoding process should infer an unintentional picture loss when any of these values of frame_num pertaining to "non-existing" pictures is referred to in inter prediction or is referred to in the assignment process for reference indices. The decoding process should not infer an unintentional picture loss when a memory management control operation is applied to a frame marked as "non-existing".

注 － 如果任何一个属于"不存在"图像的frame_num被用作帧间预测参考或者参与参考索引的分配过程的话，那么解码将认为出现了非故意的图像丢失（可以认为是丢包或码流中出现误码）。当对标识为"不存在"图像帧进行缓存控制操作时，解码器不应该将其作为非故意的图像丢失！

8.2.5.3 Sliding window decoded reference picture marking process

8.2.5.3 滑动窗口解码参考图像的标识过程

This process is invoked when adaptive_ref_pic_marking_mode_flag is equal to 0.

当 adaptive_ref_pic_marking_mode_flag 等于 0 时，使用本过程。

- If the current picture is a coded field that is the second field in decoding order of a complementary reference field pair, and the first field has been marked as "used for short-term reference", the current picture is also marked as "used for short-term reference".

- Otherwise, the following applies.

- Let numShortTerm be the total number of reference frames, complementary reference field pairs and non-paired reference fields for which at least one field is marked as "used for short-term reference". Let numLongTerm be the total number of reference frames, complementary reference field pairs and non-paired reference fields for which at least one field is marked as "used for long-term reference".

- When numShortTerm + numLongTerm is equal to num_ref_frames, the condition that numShortTerm is greater than 0 shall be fulfilled, and the short-term reference frame, complementary reference field pair or non-paired reference field that has the smallest value of FrameNumWrap is marked as "unused for reference". When it is a frame or a complementary field pair, both of its fields are also marked as "unused for reference".

- 如果当前图像是一编码场（底场或顶场），按照解码顺序来说也是一互补参考场对的第二场（例如：如果第一场是顶场，那么第二场就是底场），并且该互补参考场对的第一场被标识为"用作短期参考"，那么当前图像也被标识为"用作短期参考"。

- 否则，应用下面的规则：

- 假设 numShortTerm 是参考帧，互补参考场对和非成对的参考场的数量之和，它们中至少有一场被标识为"用作短期参考"。假设 numLongTerm 是参考帧，互补参考场对和非成对的参考场的数量之和，它们中至少有一场被标识为"用作短期参考"。

- 当 numShortTerm + numLongTerm 等于 num_ref_frames 时，numShortTerm 大于 0 的条件也满足，那么具有最小 FrameNumWrap 的短期参考帧、互补参考场对或非成对的参考场 就被标识为"不用作参考"，如果是帧或互补的参考场对时，它的两个场也被标识为"不用作参考"。

8.2.5.4 Adaptive memory control decoded reference picture marking process

8.2.5.4 自适应缓存控制解码参考图像的标识过程

This process is invoked when adaptive_ref_pic_marking_mode_flag is equal to 1.

当 adaptive_ref_pic_marking_mode_flag 等于 1 时，使用本过程。

The memory_management_control_operation commands with values of 1 to 6 are processed in the order they occur in the bitstream after the current picture has been decoded. For each of these memory_management_control_operation commands, one of the processes specified in subclauses 8.2.5.4.1 to 8.2.5.4.6 is invoked depending on the value of memory_management_control_operation. The memory_management_control_operation command with value of 0 specifies the end of memory_management_control_operation commands.

当前图像解码后，根据 memory_management_control_operation 在码流中出现的顺序，逐个进行处理。根据 memory_management_control_operation 具体的值，使用 8.2.5.4.1 到 8.2.5.4.6 小节中一个处理过程。memory_management_control_operation 等于 0 说明了 缓冲区管理控制操作的结束。

Memory management control operations are applied to pictures as follows.

缓冲区管理控制操作用于下列图像：

- If field_pic_flag is equal to 0, memory_management_control_operation commands are applied to the frames or complementary reference field pairs specified.

- Otherwise (field_pic_flag is equal to 1), memory_management_control_operation commands are applied to the individual reference fields specified.

- 如果 field_pic_flag 等于 0 ，那么 memory_management_control_operation 用于 说明的帧图像或互补的参考场对。

- 否则（field_pic_flag 等于 1），memory_management_control_operation 用于说明的单独参考场。

8.2.5.4.1 Marking process of a short-term picture as "unused for reference"

8.2.5.4.1 短期图像标识为"不用作参考"的过程

This process is invoked when memory_management_control_operation is equal to 1.

当 memory_management_control_operation 等于 1 时使用本过程。

Let picNumX be specified by

picNumX 的说明如下：

$$\text{picNumX} = \text{CurrPicNum} - (\text{difference\_of\_pic\_nums\_minus1} + 1). \qquad (8\text{-}40)$$

Depending on field_pic_flag the value of picNumX is used to mark a short-term picture as “unused for reference” as follows.

根据 field_pic_flag 的值，用 picNumX 来标识一短期图像为“不用作参考” 的过程如下：

- If field_pic_flag is equal to 0, the short-term reference frame or short-term complementary reference field pair specified by picNumX and both of its fields are marked as “unused for reference”.
- Otherwise (field_pic_flag is equal to 1), the short-term reference field specified by picNumX is marked as “unused for reference”. When that reference field is part of a reference frame or a complementary reference field pair, the reference frame or complementary field pair is also marked as "unused for reference", but the marking of the other field is not changed.
- 如果 field_pic_flag 等于 0， 由 picNumX 所标识的短期参考帧或短期互补参考场对 和 它们的两个场都被标识为“不用作参考”；
- 否则（field_pic_flag 等于 1），由 picNumX 所表示的短期参考场被标识为“不用作参考”，如果该场是一参考帧或互补的参考场对的一部分，那么参考帧或互补的参考场对也被标识为“不用作参考”，但是另外一帧的标识不变。

NOTE – In this case, the marking of the other field is not changed by this invocation of this process, but will be changed by another invocation of this process, as specified in subclause 7.4.3.3.

注 － 在这种情况下，使用本处理过程时另一场的标识不会改变，但是在 7.4.3.3 的处理过程中会发生改变。

8.2.5.4.2 Marking process of a long-term picture as “unused for reference”

8.2.5.4.1 长期图像标识为“不用作参考”的过程

This process is invoked when memory_management_control_operation is equal to 2.

当 memory_management_control_operation 等于 1 时使用本过程。

Depending on field_pic_flag the value of LongTermPicNum is used to mark a long-term picture as “unused for reference” as follows.

根据 field_pic_flag 的值，用 LongTermPicNum 来标识一长期图像为“不用作参考” 的过程如下：

- If field_pic_flag is equal to 0, the long-term reference frame or long-term complementary reference field pair having LongTermPicNum equal to long_term_pic_num and both of its fields are marked as “unused for reference”.
- Otherwise (field_pic_flag is equal to 1), the long-term reference field specified by LongTermPicNum equal to long_term_pic_num is marked as “unused for reference”. When that reference field is part of a reference frame or a complementary reference field pair, the reference frame or complementary field pair is also marked as "unused for reference", but the marking of the other field is not changed.
- 如果 field_pic_flag 等于 0， 那么由 LongTermPicNum 所标识的长期参考帧或长期互补参考场对和它们的两个场都被标识为“不用作参考”；
- 否则（field_pic_flag 等于 1），由 LongTermPicNum 所标识的长期参考场被标识为“不用作参考”。如果该参考场是一参考帧或一互补参考场对的一部分，那么该参考帧或互补参考场对也被标识为“不用作参考”，但是另外一场的标识不变。

NOTE – In this case, the marking of the other field is not changed by this invocation of this process, but will be changed by another invocation of this process, as specified in subclause 7.4.3.3.

注 － 在这种情况下，使用本处理过程时另一场的标识不会改变，但是在 7.4.3.3 的处理过程中会发生改变。

8.2.5.4.3 Assignment process of a LongTermFrameIdx to a short-term reference picture

8.2.5.4.3 给一短期参考图像分配一 LongTermFrameIdx 的过程

This process is invoked when memory_management_control_operation is equal to 3.

当 memory_management_control_operation 等于 3 时使用本过程。

Given the syntax element difference_of_pic_nums_minus1, the variable picNumX is obtained as specified in subclause 8.2.5.4.1. picNumX shall refer to a frame or complementary reference field pair or non-paired reference field marked as "used for short-term reference" and not marked as "non-existing".

根据给定的语法元素 difference_of_pic_nums_minus1，使用 8.2.5.4.1 小节中的方法计算出变量 picNumX 的值，picNumX 表示标识为“用作短期参考”但没有“不存在” 标识的帧或互补参考场对或非成对的参考场。

When LongTermFrameIdx equal to long_term_frame_idx is already assigned to a long-term reference frame or a long-term complementary reference field pair, that frame or complementary field pair and both of its fields are marked as "unused for reference". When LongTermFrameIdx is already assigned to a non-paired reference field, and the field is not the complementary field of the picture specified by picNumX, that field is marked as “unused for reference”.

当等于 long_term_frame_idx 的 LongTermFrameIdx 已经被分配给一长期参考帧或一长期互补参考场对，那么该帧或互补场对和它们的两个场都被标识为“不用作参考”。当 LongTermFrameIdx 被分配给了一非成对的参考场并且这个场也不是 picNumX图像中的互补场，那么这个场也被标识为“不用作参考”。

Depending on field_pic_flag the value of LongTermFrameIdx is used to mark a picture from "used for short-term reference" to "used for long-term reference" as follows.

根据 field_pic_flag 的值，LongTermFrameIdx 的值将被用来标识一图像由“用作短期参考”转变为 “用作长期参考”。

- If field_pic_flag is equal to 0, the marking of the short-term reference frame or short-term complementary reference field pair specified by picNumX and both of its fields are changed from "used for short-term reference" to "used for long-term reference" and assigned LongTermFrameIdx equal to long_term_frame_idx.

- Otherwise (field_pic_flag is equal to 1), the marking of the short-term reference field specified by picNumX is changed from "used for short-term reference" to "used for long-term reference" and assigned LongTermFrameIdx equal to long_term_frame_idx.

- 如果 field_pic_flag 等于 0 ，那么短期参考帧或短期互补参考场对 picNumX 的标识将从“用作短期参考”转变为“用作长期参考”，并且将 long_term_frame_idx 分配给 LongTermFrameIdx；

- 否则（field_pic_flag 等于 1），短期参考场 picNumX 的标识将从“用作短期参考”转变为“用作长期参考”，并且将 long_term_frame_idx 分配给 LongTermFrameIdx；

8.2.5.4.4 Decoding process for MaxLongTermFrameIdx

8.2.5.4.4 MaxLongTermFrameIdx 的解码过程

This process is invoked when memory_management_control_operation is equal to 4.

当 memory_management_control_operation 等于 4 时使用本过程。

All pictures for which LongTermFrameIdx is greater than max_long_term_frame_idx_plus1 – 1 and that are marked as "used for long-term reference" shall be marked as “unused for reference”.

所有 LongTermFrameIdx 大于max_long_term_frame_idx_plus1 － 1 并且被标识为“用作长期参考”的图像将被标识为“不用作参考”。

The variable MaxLongTermFrameIdx is derived as follows.

变量 MaxLongTermFrameIdx 的计算方法如下：

- If max_long_term_frame_idx_plus1 is equal to 0, MaxLongTermFrameIdx shall be set equal to “no long-term frame indices”.

- Otherwise (max_long_term_frame_idx_plus1 is greater than 0), MaxLongTermFrameIdx shall be set equal to max_long_term_frame_idx_plus1 – 1.

- 如果 max_long_term_frame_idx_plus1 等于 0 ，那么 MaxLongTermFrameIdx 将被设置成“没有长期帧索引”。

- 否则（max_long_term_frame_idx_plus1 大于 0），那么 MaxLongTermFrameIdx 将被设置成 max_long_term_frame_idx_plus1 – 1 。

NOTE – The memory_management_control_operation command equal to 4 can be used to mark long-term reference pictures as “unused for reference”. The frequency of transmitting max_long_term_frame_idx_plus1 is not specified by this Recommendation | International Standard. However, the encoder should send a memory_management_control_operation command equal to 4 upon receiving an error message, such as an intra refresh request message.

注 － memory_management_control_operation 等于 4 可以被用来标识长期参考图像为“不用作参考”，在本协议中传输 memory_management_control_operation的频率没有做明确的说明。但是在码流发生错误时，如帧内刷新需求的消息时，编码器应该发送等于 4 的 memory_management_control_operation。

8.2.5.4.5 Marking process of all reference pictures as “unused for reference” and setting MaxLongTermFrameIdx to “no long-term frame indices”

8.2.5.4.5 标识所有的参考图像为“不用作参考”和设置 MaxLongTermFrameIdx 为“没有长期帧索引”的过程

This process is invoked when memory_management_control_operation is equal to 5.

当 memory_management_control_operation 等于 5 时使用本过程。

All reference pictures are marked as “unused for reference” and the variable MaxLongTermFrameIdx is set equal to “no long-term frame indices”.

所有的参考的图像被标识为“不用作参考”，并且设置变量 MaxLongTermFrameIdx 为“没有长期帧索引”。

8.2.5.4.6 Process for assigning a long-term frame index to the current picture

8.2.5.4.6 给当前图像分配一长期帧索引的过程

This process is invoked when memory_management_control_operation is equal to 6.

当 memory_management_control_operation 等于 6 时使用本过程。

When LongTermFrameIdx is already assigned to a long-term reference frame or a long-term complementary reference field pair, that frame or complementary field pair and both of its fields are marked as "unused for reference". When LongTermFrameIdx is already assigned to a non-paired reference field, and the field is not the complementary field of the current picture, that field is marked as “unused for reference”.

当 LongTermFrameIdx 已经被分配给一长期参考帧或一长期互补参考场对，那么该帧或互补场对和它们的两场将被标识为“不用作参考”。当 LongTermFrameIdx 已经被分配给一非成对的参考场并且这个场也不是当前图像的互补场，那么该场也被标识为“不用作参考”。

The current picture is marked as "used for long-term reference" and assigned LongTermFrameIdx equal to long_term_frame_idx.

当前图像被标识为“用作长期参考”，并且将 long_term_frame_idx 分配给 LongTermFrameIdx 。

When field_pic_flag is equal to 0, both its fields are also marked as "used for long-term reference" and assigned LongTermFrameIdx equal to long_term_frame_idx.

当 field_pic_flag 等于 0 时，那么它的两个场也被设置成“用作长期参考”，并且将 long_term_frame_idx 分配给 LongTermFrameIdx。

When field_pic_flag is equal to 1 and the current picture is a second (in decoding order) field of a complementary reference field pair, the pair is also marked as "used for long-term reference" and assigned LongTermFrameIdx equal to long_term_frame_idx.

当 field_pic_flag 等于 1 时，并且按照解码顺序当前图像是一互补参考场对的第二场，那么该互补场对将被标识为“用作长期参考”，同时也将 long_term_frame_idx 分配给 LongTermFrameIdx 。

8.3 Intra prediction process

8.3 帧内预测过程

This process is invoked for I and SI macroblock types.

对于 I 和 SI 的宏块类型，使用本过程

Inputs to this process are constructed samples prior to the deblocking filter process from neighbouring macroblocks and for Intra_4x4 prediction mode, the associated values of Intra4x4PredMode from neighbouring macroblocks.

本过程的输入是来自相邻宏块的环路滤波之前的重建象素值和 Intra_4x4 预测模式，以及相邻宏块的 Intra4x4PredMode 的值。

Outputs of this process are

本过程的输出是：

- If mb_type is not equal to I_PCM, the Intra prediction samples of components of the macroblock or in case of the Intra_4x4 prediction process for luma samples, the outputs are 4x4 luma sample arrays as part of the 16x16 luma array of prediction samples of the macroblock.
- Otherwise (mb_type is equal to I_PCM), constructed macroblock samples prior to the deblocking filter process.

如果 mb_type 不等于 I_PCM（我觉得英文原文有问题），输出是4x4的亮度数组，是16x16宏块的一部分。

否则（mb_type等于 I_PCM），输出的是环路滤波之前的重建宏块值。

Depending on the value of mb_type the following applies.

根据 mb_type 的值，使用下面的步骤：

- If mb_type is equal to I_PCM, the process specified in subclause 8.3.4 is invoked.
- 如果 mb_type 等于 I_PCM， 那么使用 8.3.4 小节中的处理过程。
- Otherwise (mb_type is not equal to I_PCM), the following applies.
- 否则（mb_type 不等于 I_PCM），使用下面的步骤：
  - The decoding processes for Intra prediction modes are described for the luma component as follows.
  - 亮度分量帧内预测模式的解码过程描述如下：
    - If the macroblock prediction mode is equal to Intra_4x4, the specification in subclause 8.3.1 applies.
    - Otherwise (the macroblock prediction mode is equal to Intra_16x16), the specification in subclause 8.3.2 applies.
    - 如果宏块的预测模式等于 Intra_4x4， 那么使用 8.3.1 小节中的过程。
    - 否则（宏块的预测模式等于 Intra_16x16），那么使用 8.3.2 小节中的过程。
  - The decoding processes for Intra prediction modes for the chroma components are described in subclause 8.3.3.
  - 8.3.3 小节描述了色差分量帧内预测的解码过程。

Samples used in the Intra prediction process shall be sample values prior to alteration by any deblocking filter operations.

帧内预测过程使用的象素都是环路滤波操作之前的象素。

8.3.1 Intra_4x4 prediction process for luma samples

8.3.1 亮度分量 Intra_4x4 的预测过程

This process is invoked when the macroblock prediction mode is equal to Intra_4x4.

当宏块的预测模式是Intra_4x4，使用本过程。

Inputs to this process are constructed luma samples prior to the deblocking filter process from neighbouring macroblocks and the associated values of Intra4x4PredMode from the neighbouring macroblocks or macroblock pairs.

本过程的输入是相邻宏块的环路滤波之前的重建亮度值和对应的相邻宏块或宏块对的 Intra4x4PredMode 的值。

Outputs of this process are 4x4 luma sample arrays as part of the 16x16 luma array of prediction samples of the macroblock $pred_L$.

本过程的输出是 4×4的亮度象素值，它是 16×16亮度宏块 predL 预测值的一部分。

The luma component of a macroblock consists of 16 blocks of 4x4 luma samples. These blocks are inverse scanned using the 4x4 luma block inverse scanning process as specified in subclause 6.4.3.

每个宏块的亮度分量由16个4×4的亮度块组成，这些块使用 6.4.3小节中的 4×4亮度块反扫描过程进行反向扫描。

For the all 4x4 luma blocks of the luma component of a macroblock with luma4x4BlkIdx = 0..15, the variable Intra4x4PredMode[luma4x4BlkIdx] is derived as specified in subclause 8.3.1.1.

对于一个宏块亮度分量的所有 4×4 块（它们的索引是 luma4x4BlkIdx = 0..15），变量 Intra4x4PredMode[luma4x4BlkIdx] 的计算方法如 8.3.1.1 小节中描述的那样。

For the each luma block of 4x4 samples indexed using luma4x4BlkIdx = 0..15,

1. The Intra_4x4 sample prediction process in subclause 8.3.1.2 is invoked with luma4x4BlkIdx and constructed samples prior (in decoding order) to the deblocking filter process from adjacent luma blocks as the input and the output are the Intra_4x4 luma prediction samples $pred4x4_L[x, y]$ with x, y = 0..3.
2. The position of the upper-left sample of a 4x4 luma block with index luma4x4BlkIdx inside the current macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (xO, yO) and x, y = 0..3.

$$pred_L[\ xO + x, yO + y\ ] = pred4x4_L[\ x, y\ ] \quad (8\text{-}41)$$

3. The transform coefficient decoding process and picture construction process prior to deblocking filter process in subclause 8.5 is invoked with $pred_L$ and luma4x4BlkIdx as the input and the constructed samples for the current 4x4 luma block S'_L as the output.

对于每一个索引为 luma4x4BlkIdx = 0..15 的 4×4亮度块，

1．用 luma4x4BlkIdx 和相邻亮度块的环路滤波之前的重建象素值作为输入，使用 8.3.1.2 小节中的 Intra_4x4 预测过程，输出即为 Intra_4x4 的亮度预测值 $pred4x4_L[\ x, y\ ]$， x, y = 0..3。

2．用 luma4x4BlkIdx 作为输入，使用 6.4.3 小节中的反向 4×4 亮度块扫描过程，输出为(xO, yO)，那么当前宏块中索引为luma4x4BlkIdx 的 4×4 亮度块的左上角象素的坐标位置为：

$$pred_L[\ xO + x, yO + y\ ] = pred4x4_L[\ x, y\ ] \quad (8\text{-}41)$$

3．然后用 $pred_L$ 和 luma4x4BlkIdx 作为输入，使用 8.5 小节中的环路滤波之前变换系数的解码过程和图像的重建过程，输出即为当前 4×4 亮度块S'_L 的重建象素值。

8.3.1.1 Derivation process for the Intra4x4PredMode

8.3.1.1 Intra4x4PredMode 的解码过程

Inputs to this process are the index of the 4x4 luma block luma4x4BlkIdx and variable arrays Intra4x4PredMode that are previously (in decoding order) derived for adjacent macroblocks.

本过程的输入是 4×4 亮度块 luma4x4BlkIdx 的索引和已经解码的相邻宏块的 Intra4x4PredMode变量数组（按解码顺序）。

Output of this process is the variable Intra4x4PredMode[luma4x4BlkIdx].

本过程的输出是变量Intra4x4PredMode[luma4x4BlkIdx]。

Table 8-2 specifies the values for Intra4x4PredMode[luma4x4BlkIdx] and the associated names.

表 8－2 Intra4x4PredMode[luma4x4BlkIdx]的值及相对应的名称

Table 8-2 – Specification of Intra4x4PredMode[luma4x4BlkIdx] and associated names

Intra4x4PredMode[luma4x4BlkIdx]	Name of Intra4x4PredMode[luma4x4BlkIdx]
0	Intra_4x4_Vertical (prediction mode)
1	Intra_4x4_Horizontal (prediction mode)
2	Intra_4x4_DC (prediction mode)
3	Intra_4x4_Diagonal_Down_Left (prediction mode)
4	Intra_4x4_Diagonal_Down_Right (prediction mode)
5	Intra_4x4_Vertical_Right (prediction mode)
6	Intra_4x4_Horizontal_Down (prediction mode)
7	Intra_4x4_Vertical_Left (prediction mode)
8	Intra_4x4_Horizontal_Up (prediction mode)

Intra4x4PredMode[luma4x4BlkIdx] labelled 0, 1, 3, 4, 5, 6, 7, and 8 represent directions of predictions as illustrated in Figure 8-1.

等于 0, 1, 3, 4, 5, 6, 7, 和 8 的 Intra4x4PredMode[luma4x4BlkIdx] 表示了 Figure 8-1 中所示的预测方向。

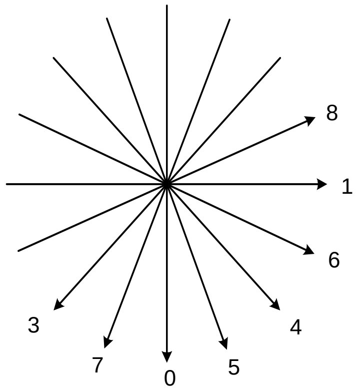


Figure 8-1 – Intra_4x4 prediction mode directions (informative)

Let intra4x4PredModeA and intra4x4PredModeB be variables that specify the intra prediction modes of neighbouring 4x4 luma blocks.

假设变量 intra4x4PredModeA 和 intra4x4PredModeB 为相邻 4x4 亮度块的帧内预测模式。

Intra4x4PredMode[luma4x4BlkIdx] is derived as follows.

Intra4x4PredMode[luma4x4BlkIdx] 的计算如下：

- The process specified in subclause 6.4.7.3 is invoked with luma4x4BlkIdx given as input and the output is assigned to mbAddrA, luma4x4BlkIdxA, mbAddrB, and luma4x4BlkIdxB.
- 将 luma4x4BlkIdx 作为输入，使用 6.4.7.3 小节中的处理过程，然后再将输出赋给 mbAddrA, luma4x4BlkIdxA, mbAddrB, 和 luma4x4BlkIdxB。
- The variable dcOnlyPredictionFlag is derived as follows.
  - If one of the following conditions is true, dcOnlyPredictionFlag is set equal to 1
    - the macroblock with address mbAddrA is not available
    - the macroblock with address mbAddrB is not available
    - the macroblock with address mbAddrA is available and coded in Inter prediction mode and constrained_intra_pred_flag is equal to 1
    - the macroblock with address mbAddrB is available and coded in Inter prediction mode and constrained_intra_pred_flag is equal to 1
  - Otherwise, dcOnlyPredictionFlag is set equal to 0.
- 变量 dcOnlyPredictionFlag 的计算如下：
  - 如果下面的条件之一成立，那么 dcOnlyPredictionFlag 将被设置成 1。
    - 地址为 mbAddrA 的宏块不可用；
    - 地址为 mbAddrB 的宏块不可用；
    - 地址为 mbAddrA的宏块可用，但是该宏块是帧间编码方式，并且constrained_intra_pred_flag等于1
    - 地址为 mbAddrB的宏块可用，但是该宏块是帧间编码方式，并且constrained_intra_pred_flag等于1
  - 否则，dcOnlyPredictionFlag 将被设置成 0。
- For N being either replaced by A or B, the variables intra4x4PredModeN are derived as follows.
  - If dcOnlyPredictionFlag is equal to 1 or the macroblock with address mbAddrN is not coded in Intra_4x4 macroblock prediction mode, intra4x4PredModeN is set equal to 2 (Intra_4x4_DC prediction mode).

- Otherwise (dcOnlyPredictionFlag is equal to 0 and the macroblock with address mbAddrN is coded in Intra_4x4 macroblock prediction mode), intra4x4PredModeN is set equal to Intra4x4PredMode[luma4x4BlkIdxN], where Intra4x4PredMode is the variable array assigned to the macroblock mbAddrN.

- 变量 intra4x4PredModeN 的计算如下， N 可能被 A 或 B 替代
  - 如果 dcOnlyPredictionFlag 等于 1 或者 地址为 mbAddrN 的宏块不是 Intra_4x4 预测模式，那么 intra4x4PredModeN 被设置成等于 2（Intra_4x4_DC 预测模式）；
  - 否则（dcOnlyPredictionFlag 等于 0 或者 地址为 mbAddrN 的宏块是 Intra_4x4 预测模式），那么 intra4x4PredModeN 被设置成 Intra4x4PredMode[luma4x4BlkIdxN]，这里 Intra4x4PredMode 是地址为 mbAddrN的宏块的数组变量。

- Intra4x4PredMode[luma4x4BlkIdx] is derived by applying the following procedure.
- Intra4x4PredMode[luma4x4BlkIdx]的计算如下：

```
predIntra4x4PredMode = Min( intra4x4PredModeA, intra4x4PredModeB )
if( prev_intra4x4_pred_mode_flag[ luma4x4BlkIdx ] )
    Intra4x4PredMode[ luma4x4BlkIdx ] = predIntra4x4PredMode
else                                                                          (8-42)
    if( rem_intra4x4_pred_mode[ luma4x4BlkIdx ] < predIntra4x4PredMode )
        Intra4x4PredMode[ luma4x4BlkIdx ] = rem_intra4x4_pred_mode[ luma4x4BlkIdx ]
    else
        Intra4x4PredMode[ luma4x4BlkIdx ] = rem_intra4x4_pred_mode[ luma4x4BlkIdx ] + 1
```

（注这里的加 1 ，应该做一个简单的说明）

8.3.1.2 Intra_4x4 sample prediction

8.3.1.2 Intra_4x4 象素值的预测

This process is invoked for each 4x4 luma block of a macroblock with prediction mode equal to Intra_4x4 followed by the transform decoding process and picture construction process prior to deblocking for each 4x4 luma block.

预测模式等于 Intra_4x4 宏块的 4x4 亮度块的解码使用这个过程，然后再做环路滤波之前的变换解码过程和图像重建过程。(这里最好画个流程图)

Inputs to this process are the index of the 4x4 luma block with index luma4x4BlkIdx and constructed samples prior (in decoding order) to the deblocking filter process from adjacent luma blocks.

本过程的输入是索引为 luma4x4BlkIdx 的亮度块和相邻亮度块的环路滤波之前的重建象素值。

Output of this process are the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 for the 4x4 luma block with index luma4x4BlkIdx.

本过程的输出是索引为 luma4x4BlkIdx 的4×4亮度块的预测象素值 pred4x4L[x, y]， x, y = 0..3。

The position of the upper-left sample of a 4x4 luma block with index luma4x4BlkIdx inside the current macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (xO, yO).

当前宏块中索引为 luma4x4BlkIdx 的 4×4亮度块的左上角象素的位置由 6.4.3 小节中的反向 4×4 亮度块扫描过程计算得到，该过程的输入是 luma4x4BlkIdx，输出被赋给 (xO, yO)。

The 13 neighbouring samples p[x, y] that are constructed luma samples prior to deblocking, with x = -1, y = -1..3 and x = 0..7, y = -1, are derived as follows.

环路滤波之前的 13 个相邻重建象素值（x = -1, y = -1..3 和 x = 0..7, y = -1）的计算如下：

– The luma location (xN, yN) is specified by

– 亮度位置 (xN, yN) 的说明如下：

$$xN = xO + x \tag{8-43}$$

$$yN = yO + y \tag{8-44}$$

– The derivation process for neighbouring locations in subclause 6.4.8 is invoked for luma locations with (xN, yN) as input and mbAddrN and (xW, yW) as output.

– 将 (xN, yN) 作为输入，使用 6.4.8小节中的关于相邻位置的处理过程，输出为相邻象素的mbAddrN and 相应的坐标位置(xW, yW)。

– If any of the following conditions is true, the sample p[x, y] is marked as “not available for Intra_4x4 prediction”

  - mbAddrN is not available,
  - the macroblock mbAddrN is coded in Inter prediction mode and constrained_intra_pred_flag is equal to 1.
  - the macroblock mbAddrN has mb_type equal to SI and constrained_intra_pred_flag is equal to 1 and the current macroblock does not have mb_type equal to SI.
  - x is greater than 3 and luma4x4BlkIdx is equal to 3 or 11

– 如果满足下面的任何一个条件，那么象素 p[x, y] 就被标识为“不可用作 Intra_4x4 预测”。

  - mbAddrN 不可用。
  - 地址为 mbAddrN 的宏块使用的是帧间预测模式，并且 constrained_intra_pred_flag 等于 1。
  - 地址为mbAddrN 的宏块的 mb_type 是 SI， constrained_intra_pred_flag 也等于 1，并且当前宏块的 mb_type 不是 SI。
  - x 大于 3 并且 luma4x4BlkIdx 等于 3 或 11。

– Otherwise, the sample p[x, y] is marked as “available for Intra_4x4 prediction” and the following applies

  - The luma sample at luma location (xW, yW) inside the macroblock mbAddrN is assigned to p[x, y].

– 否则，象素 p[x, y] 被标识为“可用作 Intra_4x4 预测”，并且使用下面的过程

  - 在地址为mbAddrN 的宏块内位置为 (xW, yW) 的亮度象素值赋给 p[x, y]。

When samples p[x, -1], with x = 4..7 are marked as “not available for Intra_4x4 prediction,” and the sample p[3, -1] is marked as “available for Intra_4x4 prediction,” the sample value of p[3, -1] is substituted for sample values p[x, -1], with x = 4..7 and samples p[x, -1], with x = 4..7 are marked as “available for Intra_4x4 prediction”.

NOTE – Each block is assumed to be reconstructed into a frame prior to decoding of the next block.

当象素p[x, -1]（ x = 4..7 ）被标识为“不可用作Intra_4x4预测”并且象素p[3, -1]被标识为“可用作Intra_4x4预测”时，那么象素 p[x, -1]（ x = 4..7 ）的值将由p[3, -1]来替代，并都被标识为“可用作Intra_4x4预测”。

NOTE – Each block is assumed to be reconstructed into a frame prior to decoding of the next block.
注 － 在解码下一个块之前，每一个块都被假设重建成帧。

Depending on Intra4x4PredMode[luma4x4BlkIdx], one of the Intra_4x4 prediction modes specified in subclauses 8.3.1.2.1 to 8.3.1.2.9 shall be used.

根据 Intra4x4PredMode[luma4x4BlkIdx] 的值，下面 8.3.1.2.1 to 8.3.1.2.9 分别说明了 Intra_4x4 的一种预测模式。

8.3.1.2.1 Specification of Intra_4x4_Vertical prediction mode

8.3.1.2.1 Intra_4x4_Vertical 预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 0.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 0 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..3 are marked as “available for Intra_4x4 prediction”.

只用当象素 p[x, -1]（x = 0..3）被标识为“可用作 Intra_4x4 预测”时才能使用该模式。

The values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:

预测象素 pred4x4L[x, y]（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = p[x, -1], \text{ with } x, y = 0..3 \quad (8\text{-}45)$$

8.3.1.2.2 Specification of Intra_4x4_Horizontal prediction mode

8.3.1.2.2 Intra_4x4_Horizontal预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 1.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 1 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[-1, y], with y = 0..3 are marked as “available for Intra_4x4 prediction”.

只用当象素 p[-1, y]（y = 0..3）被标识为“可用作 Intra_4x4 预测”时才能使用该模式。

The values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:

预测象素 pred4x4L[x, y]（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = p[-1, y], \text{ with } x,y = 0..3 \quad (8\text{-}46)$$

8.3.1.2.3 Specification of Intra_4x4_DC prediction mode

8.3.1.2.3 Intra_4x4_DC预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 2.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 2 时，使用这个 Intra_4x4 预测模式。

The values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:

预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

- If all samples p[x, -1], with x = 0..3 and p[-1, y], with y = 0..3 are marked as “available for Intra_4x4 prediction”, the values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:
- 如果所有象素 p[x, -1]（x = 0..3）和 p[-1, y]（y = 0..3）都被标识为“可用作 Intra_4x4 预测”，那么预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 4) >> 3 \quad (8\text{-}47)$$

- If samples p[x, -1], with x = 0..3 are marked as “not available for Intra_4x4 prediction” and p[-1, y], with y = 0..3 are marked as “available for Intra_4x4 prediction”, the values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:
- 如果象素 p[x, -1]（x = 0..3）被标识为“不可用作 Intra_4x4 预测”，而象素 p[-1, y]（y = 0..3）被标识为“可用作 Intra_4x4 预测”，那么预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = (p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 2) >> 2 \quad (8\text{-}48)$$

- If samples p[-1, y], with y = 0..3 are marked as “not available for Intra_4x4 prediction” and p[x, -1], with x = 0 .. 3 are marked as “available for Intra_4x4 prediction”, the values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0 .. 3 are derived as follows:
- 如果象素 p[x, -1]（x = 0..3）被标识为“可用作 Intra_4x4 预测”，而象素 p[-1, y]（y = 0..3）被标识为“不可用作 Intra_4x4 预测”，那么预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + 2) >> 2 \quad (8\text{-}49)$$

- Otherwise (all samples p[x, -1], with x = 0..3 and p[-1, y], with y = 0..3 are marked as “not available for Intra_4x4 prediction”), the values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:
- 否则（所有象素 p[x, -1]（x = 0..3）和 p[-1, y]（y = 0..3）都被标识为“不可用作 Intra_4x4 预测”），预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

$$pred4x4_L[x, y] = 128 \quad (8\text{-}50)$$

NOTE – A block can always be predicted using this mode.
注 — 一个块也许只能用这种模式

8.3.1.2.4 Specification of Intra_4x4_Diagonal_Down_Left prediction mode

8.3.1.2.4 Intra_4x4_Diagonal_Down_Left预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 3.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 3 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..7 are marked as “available for Intra_4x4 prediction”.

只用当象素 p[x, -1]（x = 0..7）被标识为“可用作 Intra_4x4 预测”时才使用这个模式。

The values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:

预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

- If x is equal to 3 and y is equal to 3,
- 如果 x 等于 3 并且 y 也等于 3，

$$pred4x4_L[x, y] = (p[6, -1] + 3 * p[7, -1] + 2) >> 2 \quad (8-51)$$

- Otherwise (x is not equal to 3 or y is not equal to 3),
- 否则（x 不等于 3 或者 y 不等于 3）

$$pred4x4_L[x, y] = (p[x + y, -1] + 2 * p[x + y + 1, -1] + p[x + y + 2, -1] + 2) >> 2 \quad (8-52)$$

8.3.1.2.5 Specification of Intra_4x4_Diagonal_Down_Right prediction mode

8.3.1.2.5 Intra_4x4_Diagonal_Down_Right预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 4.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 4 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..3 and p[-1, y] with y = -1..3 are marked as “available for Intra_4x4 prediction”.

只有当象素 p[x, -1]（x = 0..3）和象素p[-1, y]（y = -1..3）都被标识为“可用作 Intra_4x4 预测”时才能使用这个模式。

The values of the prediction samples $pred4x4_L[x, y]$, with x, y = 0..3 are derived as follows:

预测象素 $pred4x4_L[x, y]$（x, y = 0..3）的值计算如下：

- If x is greater than y,
- 如果 x 大于 y，

$$pred4x4_L[x, y] = (p[x - y - 2, -1] + 2 * p[x - y - 1, -1] + p[x - y, -1] + 2) >> 2 \quad (8-53)$$

- If x is less than y,
- 如果 x 小于 y，

$$pred4x4_L[x, y] = (p[-1, y - x - 2] + 2 * p[-1, y - x - 1] + p[-1, y - x] + 2) >> 2 \quad (8-54)$$

- Otherwise (x is equal to y),
- 否则（x 等于 y），

$$pred4x4_L[x, y] = (p[0, -1] + 2 * p[-1, -1] + p[-1, 0] + 2) >> 2 \quad (8-55)$$

8.3.1.2.6 Specification of Intra_4x4_Vertical_Right prediction mode

8.3.1.2.6 Intra_4x4_Vertical_Right预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 5.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 5 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..3 and p[-1, y] with y = -1..3 are marked as “available for Intra_4x4 prediction”.

只有当象素 p[x, -1]（x = 0..3）和 象素p[-1, y] （ y = -1..3）都被标识为“可用作 Intra_4x4 预测”时才使用这个模式。

Let the variable zVR be set equal to 2 * x – y.

假设变量 zVR 被设置成 2 * x – y。

The values of the prediction samples $pred4x4_L$[x, y], with x, y = 0..3 are derived as follows:

预测象素 $pred4x4_L$[x, y]（x, y = 0..3）的值计算如下：

- If zVR is equal to 0, 2, 4, or 6,
- 如果 zVR 等于 0, 2, 4, 或 6,

$$pred4x4_L[x, y] = (p[x - (y >> 1) - 1, -1] + p[x - (y >> 1), -1] + 1) >> 1 \quad (8\text{-}56)$$

- If zVR is equal to 1, 3, or 5,
- 如果 zVR 等于 1, 3, 或5,

$$pred4x4_L[x, y] = (p[x - (y >> 1) - 2, -1] + 2 * p[x - (y >> 1) - 1, -1] + p[x - (y >> 1), -1] + 2) >> 2 \quad (8\text{-}57)$$

- If zVR is equal to -1,
- 如果 zVR 等于 －1，

$$pred4x4_L[x, y] = (p[-1, 0] + 2 * p[-1, -1] + p[0, -1] + 2) >> 2 \quad (8\text{-}58)$$

- Otherwise (zVR is equal to -2 or -3),
- 否则（zVR 等于-2 或 -3 ），

$$pred4x4_L[x, y] = (p[-1, y - 1] + 2 * p[-1, y - 2] + p[-1, y - 3] + 2) >> 2 \quad (8\text{-}59)$$

8.3.1.2.7 Specification of Intra_4x4_Horizontal_Down prediction mode

8.3.1.2.7 Intra_4x4_Horizontal_Down预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 6.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 6 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..3 and p[-1, y] with y = -1..3 are marked as “available for Intra_4x4 prediction”.

只有当象素p[x, -1] （x = 0..3）和 p[-1, y] (y = -1..3)都被标识为“可用作 Intra_4x4 预测”时才能使用这个模式。

Let the variable zHD be set equal to 2 * y – x.

假设变量 zHD 被设置成 2 * y – x。

The values of the prediction samples $pred4x4_L$[x, y], with x, y = 0..3 are derived as follows:

预测象素 pred4x4L[x, y]（x, y = 0..3）的值计算如下：

- If zHD is equal to 0, 2, 4, or 6,
- 如果 zHD 等于 0, 2, 4, 或 6,

$$pred4x4_L[x, y] = (p[-1, y -(x >> 1) - 1] + p[-1, y - (x >> 1)] + 1) >> 1 \quad (8\text{-}60)$$

- If zHD is equal to 1, 3, or 5,
- 如果 zHD 等于 1, 3, 或 5,

$$pred4x4_L[x, y] = (p[-1, y - (x >> 1) - 2] + 2 * p[-1, y - (x >> 1) - 1] + p[-1, y - (x >> 1)] + 2) >> 2 \quad (8\text{-}61)$$

- If zHD is equal to -1,
- 如果 zHD 等于 -1,

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,-1, 0\,] + 2 * p[\,-1, -1\,] + p[\,0, -1\,] + 2\,) >> 2 \quad (8\text{-}62)$$

- Otherwise (zHD is equal to -2 or -3),
- 否则（zHD 等于 -2 或 -3），

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,x - 1, -1\,] + 2 * p[\,x - 2, -1\,] + p[\,x - 3, -1\,] + 2\,) >> 2 \quad (8\text{-}63)$$

8.3.1.2.8 Specification of Intra_4x4_Vertical_Left prediction mode

8.3.1.2.8 Intra_4x4_Vertical_Left预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 7.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 7 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[x, -1] with x = 0..7 are marked as “available for Intra_4x4 prediction”.

只有当象素 p[x, -1]（x = 0..7）被标识为“可用作 Intra_4x4 预测”时才能使用这个模式。

The values of the prediction samples pred4x4_L[x, y], with x, y = 0..3 are derived as follows:

预测象素 pred4x4_L[x, y]（x, y = 0..3）的值计算如下：

- If y is equal to 0 or 2,
- 如果 y 等于 0 或 2，

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,x + (\,y >> 1\,), -1\,] + p[\,x + (\,y >> 1\,) + 1, -1\,] + 1) >> 1 \quad (8\text{-}64)$$

- Otherwise (y is equal to 1 or 3),
- 否则（y 等于 1 或 3），

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,x + (\,y >> 1\,), -1\,] + 2 * p[\,x + (\,y >> 1\,) + 1, -1\,] + p[\,x + (\,y >> 1\,) + 2, -1\,] + 2\,) >> 2 \quad (8\text{-}65)$$

8.3.1.2.9 Specification of Intra_4x4_Horizontal_Up prediction mode

8.3.1.2.9 Intra_4x4_Horizontal_Up预测模式的说明

This Intra_4x4 prediction mode shall be used when Intra4x4PredMode[luma4x4BlkIdx] is equal to 8.

当 Intra4x4PredMode[luma4x4BlkIdx] 等于 8 时，使用这个 Intra_4x4 预测模式。

This mode shall be used only when the samples p[-1, y] with y = 0..3 are marked as “available for Intra_4x4 prediction”.

只用当 p[-1, y]（y = 0..3）被标识为“可用作 Intra_4x4 预测”时才能使用这个模式。

Let the variable zHU be set equal to x + 2 * y.

假设变量 zHU 被设置成 x + 2 * y。

The values of the prediction samples pred4x4_L[x, y], with x, y = 0..3 are derived as follows:

预测象素 pred4x4L[x, y]（x, y = 0..3）的值计算如下：

- If zHU is equal to 0, 2, or 4
- 如果 zHU 等于 0, 2, 或 4

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,-1, y + (\,x >> 1\,)\,] + p[\,-1, y + (\,x >> 1\,) + 1\,] + 1\,) >> 1 \quad (8\text{-}66)$$

- If zHU is equal to 1 or 3
- 如果 zHU 等于 1 或 3

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,-1, y + (\,x >> 1\,)\,] + 2 * p[\,-1, y + (\,x >> 1\,) + 1\,] + p[\,-1, y + (\,x >> 1\,) + 2\,] + 2\,) >> 2 \quad (8\text{-}67)$$

- If zHU is equal to 5,
- 如果 zHU 等于 5,

$$\text{pred4x4}_L[\,x, y\,] = (\,p[\,-1, 2\,] + 3 * p[\,-1, 3\,] + 2\,) >> 2 \quad (8\text{-}68)$$

- Otherwise (zHU is greater than 5),
- 否则（zHU 大于 5），

$$\text{pred4x4}_L[\,x, y\,] = p[\,-1, 3\,] \quad (8\text{-}69)$$

8.3.2 Intra_16x16 prediction process for luma samples

8.3.2 亮度分量 Intra_16x16 预测模式的处理过程

This process is invoked when the macroblock prediction mode is equal to Intra_16x16. It specifies how the Intra prediction luma samples for the current macroblock are derived.

当宏块的预测模式等于 Intra_16x16 时使用本过程，本过程说明了如何计算当前宏块亮度分量的预测值。

Input to this process are constructed samples prior to the deblocking process from neighbouring luma blocks (if available).

本过程的输入是相邻亮度块（如果可用）的环路滤波之前的重建象素值。

Outputs of this process are Intra prediction luma samples for the current macroblock $\text{pred}_L[\,x, y\,]$.

本过程的输出是当前宏块亮度分量的帧内预测值 predL[x, y]。

The 33 neighbouring samples p[x, y] that are constructed luma samples prior to deblocking, with x = -1, y = -1..15 and with x = 0..15, y = -1, are derived as follows.

这 33 个环路滤波之前的相邻重建亮度象素 p[x, y]（x = -1, y = -1..15 和 x = 0..15, y = -1）的值计算如下：

– The derivation process for neighbouring locations in subclause 6.4.8 is invoked for luma locations with (x, y) assigned to (xN, yN) as input and mbAddrN and (xW, yW) as output.
– If any of the following conditions is true, the sample p[x, y] is marked as “not available for Intra_16x16 prediction”
  - mbAddrN is not available,
  - the macroblock mbAddrN is coded in Inter prediction mode and constrained_intra_pred_flag is equal to 1.
  - the macroblock mbAddrN has mb_type equal to SI and constrained_intra_pred_flag is equal to 1.
– Otherwise, the sample p[x, y] is marked as “available for Intra_16x16 prediction” and the following applies
  - The luma sample at luma location (xW, yW) inside the macroblock mbAddrN is assigned to p[x, y].
— 将 (x, y) 赋给 (xN, yN) 并作为输入，使用 6.4.8 小节中的相邻位置的处理过程，输出为 mbAddrN 和 (xW, yW)。
— 如果满足下面的任何一个条件，那么象素p[x, y]将被标识为“不可用作Intra_16x16预测”
  - mbAddrN 不可用；
  - 地址为 mbAddrN 的宏块是帧间编码模式，并且 constrained_intra_pred_flag 等于 1；
  - 地址为 mbAddrN 的宏块的mb_type为 SI，并且 constrained_intra_pred_flag 等于 1（应该还有当前宏块的mb_type不为 SI）；
— 否则，象素 p[x, y] 将被标识为“可用作Intra_16x16预测”，并且使用下面的过程
  - 在地址为 mbAddrN 的宏块内位置为(xW, yW)的亮度分量的象素分配给 p[x, y]。

Let $\text{pred}_L[\,x, y\,]$ with x, y = 0..15 denote the prediction samples for the 16x16 luma block samples.

假设 $pred_L$[x, y]（ x, y = 0..15）表示 16x16 亮度块象素的预测值。

Intra_16x16 prediction modes are specified in Table 8-3.

Intra_16x16 prediction 的预测模式如表 Table 8-3 所示。

Table 8-3 – Specification of Intra16x16PredMode and associated names

Intra16x16PredMode	Name of Intra16x16PredMode
0	Intra_16x16_Vertical (prediction mode)
1	Intra_16x16_Horizontal (prediction mode)
2	Intra_16x16_DC (prediction mode)
3	Intra_16x16_Plane (prediction mode)

Depending on Intra16x16PredMode, one of the Intra_16x16 prediction modes specified in subclauses 8.3.2.1 to 8.3.2.4 shall be used.

根据 Intra16x16PredMode 的值，8.3.2.1 到 8.3.2.4 小节分别介绍了一种 Intra_16x16 的预测模式。

8.3.2.1 Specification of Intra_16x16_Vertical prediction mode

8.3.2.1 Intra_16x16_Vertical 预测模式的说明

This mode shall be used only when the samples p[x, -1] with x = 0..15 are marked as “available for Intra_16x16 prediction”.

只有当象素p[x, -1]（x = 0..15）被标识为“可用作 Intra_16x16 预测”时才使用这个模式。

$$pred_L[\,x, y\,] = p[\,x, -1\,],\ \text{with } x, y = 0..15 \tag{8-70}$$

8.3.2.2 Specification of Intra_16x16_Horizontal prediction mode

8.3.2.2 Intra_16x16_Horizontal 预测模式的说明

This mode shall be used only when the samples p[-1, y] with y = 0..15 are marked as “available for Intra_16x16 prediction”.

只有当象素 p[-1, y]（ y = 0..15）被标识为“可用作 Intra_16x16 预测”时才使用这个模式。

$$pred_L[\,x, y\,] = p[\,-1, y\,],\ \text{with } x, y = 0..15 \tag{8-71}$$

8.3.2.3 Specification of Intra_16x16_DC prediction mode

8.3.2.3 Intra_16x16_DC 预测模式的说明

Dependent on whether the neighbouring samples are marked as “available for Intra_16x16 prediction” the following applies.

根据相邻象素是否标识为“可用作 Intra_16x16 预测”，使用下面的过程。

- If all neighbouring samples p[x', -1] and p[-1, y'] used in Equation 8-72 are marked as “available for Intra_16x16 prediction”, the prediction for all luma samples in the macroblock is given by:
- 如果等式 8-72 中的相邻象素 p[x', -1] 和 p[-1, y'] 都被标识为“可用作 Intra_16x16 预测”，那么当前宏块中所有亮度象素的预测如下：

$$pred_L[\,x, y\,] = \left(\sum_{x'=0}^{15} p[x', -1] + \sum_{y'=0}^{15} p[-1, y'] + 16\right) >> 5 \ \text{ with } x, y = 0..15 \tag{8-72}$$

- If the neighbouring samples p[x', -1] are not available and the neighbouring samples p[-1, y'] are marked as “available for Intra_16x16 prediction”, the prediction for all luma samples in the macroblock is given by:
- 如果相邻象素 p[x', -1] 不可用，但是相邻象素p[-1, y']被标识为“可用作 Intra_16x16 预测”，那么当前宏块中所有亮度象素的预测如下：

$$\text{pred}_L[\,x, y\,] = (\sum_{y'=0}^{15} p[-1, y'] + 8) >> 4 \text{ with } x, y = 0..15 \quad (8\text{-}73)$$

- If the neighbouring samples p[-1, y'] are not available and the neighbouring samples p[x', -1] are marked as "available for Intra_16x16 prediction", the prediction for all luma samples in the macroblock is given by:
- 如果相邻象素 p[-1, y'] 不可用，但是相邻象素 p[x', -1]被标识为“可用作 Intra_16x16 预测”，那么当前宏块中所有亮度象素的预测如下：

$$\text{pred}_L[\,x, y\,] = (\sum_{x'=0}^{15} p[x', -1] + 8) >> 4 \text{ with } x, y = 0..15 \quad (8\text{-}74)$$

- Otherwise (none of the neighbouring samples p[x', -1] and p[-1, y'] are marked as "available for Intra_16x16 prediction"), the prediction for all luma samples in the macroblock is given by:
- 否则（相邻象素p[x', -1] 和 p[-1, y']都没有被标识为“可用作 Intra_16x16 预测”），当前宏块中所有亮度象素的预测如下：

$$\text{pred}_L[\,x, y\,] = 128 \text{ with } x, y = 0..15 \quad (8\text{-}75)$$

8.3.2.4 Specification of Intra_16x16_Plane prediction mode

8.3.2.4 Intra_16x16_Plane 预测模式的说明

This mode shall be used only when the samples p[x, -1] with x = -1..15 and p[-1, y] with y = 0..15 are marked as "available for Intra_16x16 prediction".

只有当象素 p[x, -1] （x = -1..15） 和 p[-1, y] （ y = 0..15）都被标识为“可用作 Intra_16x16 预测”时才使用这个模式。

$$\text{pred}_L[\,x, y\,] = \text{Clip1}(\,(\,a + b * (\,x - 7\,) + c * (\,y - 7\,) + 16\,) >> 5\,), \quad (8\text{-}76)$$

where:

这里：

$$a = 16 * (\,p[\,-1, 15\,] + p[\,15, -1\,]\,) \quad (8\text{-}77)$$

$$b = (\,5 * H + 32\,) >> 6 \quad (8\text{-}78)$$

$$c = (\,5 * V + 32\,) >> 6 \quad (8\text{-}79)$$

and H and V are specified in Equations 8-80 and 8-81.

等式 8-80 和 8-81 中的 H 和 V 说明如下：

$$H = \sum_{x'=0}^{7} (\,x'+1\,) * (\,p[\,8 + x', -1\,] - p[\,6 - x', -1\,]\,) \quad (8\text{-}80)$$

$$V = \sum_{y'=0}^{7} (\,y'+1\,) * (\,p[\,-1, 8 + y'\,] - p[\,-1, 6 - y'\,]\,) \quad (8\text{-}81)$$

8.3.3 Intra prediction process for chroma samples

This process is invoked for I and SI macroblock types. It specifies how the Intra prediction chroma samples for the current macroblock are derived.

对于 I 和 SI类型的宏块使用本过程。本过程说明了如何计算当前宏块的帧内预测色差分量。

Inputs to this process are constructed samples prior to the deblocking process from neighbouring chroma blocks (if available).

本过程的输入是相邻色差块（如果可用）的环路滤波之前的重建象素值。

Outputs of this process are Intra prediction chroma samples for the current macroblock pred_{Cb}[x, y] and pred_{Cr}[x, y].

本过程的输出是当前宏块的帧内预测色差象素值$pred_{Cb}[x, y]$ 和 $pred_{Cr}[x, y]$。

Both chroma blocks (Cb and Cr) of the macroblock shall use the same prediction mode. The prediction mode is applied to each of the chroma blocks separately. The process specified in this subclause is invoked for each chroma block. In the remainder of this subclause, chroma block refers to one of the two chroma blocks and the subscript C is used as a replacement of the subscript Cb or Cr.

宏块的两个色差块使用相同的预测模式，每个色差块分别使用这个预测模式。每个色差块分别使用本小节的过程。在本小节的余下部分色差块指得是两个色差块，下标 C 用来替换下标 Cb 或 Cr。

The 17 neighbouring samples p[x, y] that are constructed chroma samples prior to deblocking, with x = -1, y = -1..7 and with x = 0..7, y = -1, are derived as follows.

这 17 个环路滤波之前的相邻重建色差象素值 p[x, y]（x = -1, y = -1..7 和 x = 0..7, y = -1）的计算如下：

– The derivation process for neighbouring locations in subclause 6.4.8 is invoked for chroma locations with (x, y) assigned to (xN, yN) as input and mbAddrN and (xW, yW) as output.
– If any of the following conditions is true, the sample p[x, y] is marked as “not available for Intra chroma prediction”
  - mbAddrN is not available,
  - the macroblock mbAddrN is coded in Inter prediction mode and constrained_intra_pred_flag is equal to 1.
  - the macroblock mbAddrN has mb_type equal to SI and constrained_intra_pred_flag is equal to 1 and the current macroblock does not have mb_type equal to SI.
– Otherwise, the sample p[x, y] is marked as “available for Intra chroma prediction” and the following applies
  - The chroma sample of component C at chroma location (xW, yW) inside the macroblock mbAddrN is assigned to p[x, y].

一 将 (x, y) 赋给 (xN, yN) 并作为输入，使用 6.4.8 小节中的相邻位置的处理过程，输出为 mbAddrN 和 (xW, yW)。
一 如果满足下面的任何一个条件，那么象素p[x, y]将被标识为“不可用作Intra_16x16预测”
  - mbAddrN 不可用；
  - 地址为 mbAddrN 的宏块是帧间编码模式，并且 constrained_intra_pred_flag 等于 1；
  - 地址为 mbAddrN 的宏块的mb_type为 SI，并且 constrained_intra_pred_flag 等于 1，并且当前宏块的mb_type不是SI；
一 否则，象素 p[x, y] 将被标识为“可用作帧内色差预测”，并且使用下面的过程
  - 在地址为 mbAddrN 的宏块内位置为(xW, yW)的色差分量 C的象素分配给 p[x, y]。

Let $pred_C[x, y]$ with x, y = 0..7 denote the prediction samples for the chroma block samples.

假设 $pred_C[x, y]$（x, y = 0..7）表示色差块象素的预测值。

Intra chroma prediction modes are specified in Table 8-4.

帧内色差预测模式如表Table 8-4 所示。

Table 8-4 – Specification of Intra chroma prediction modes and associated names

intra_chroma_pred_mode	Name of intra_chroma_pred_mode
0	Intra_Chroma_DC (prediction mode)
1	Intra_Chroma_Horizontal (prediction mode)
2	Intra_Chroma_Vertical (prediction mode)
3	Intra_Chroma_Plane (prediction mode)

Depending on intra_chroma_pred_mode, one of the Intra chroma prediction modes specified in subclauses 8.3.3.1 to 8.3.3.4 shall be used.

根据 intra_chroma_pred_mode 的值，8.3.3.1 到 8.3.3.4 分别介绍了一种帧内色差预测模式。

8.3.3.1 Specification of Intra_Chroma_DC prediction mode

8.3.3.1 Intra_Chroma_DC 预测模式的说明

The values of the prediction samples $pred_C[x, y]$ with x = 0..3 and y = 0..3 are derived as follows.

预测象素 $pred_C[x, y]$（x = 0..3 和 y = 0..3）的值计算如下：

- If the samples p[x, –1] with x = 0..3 and the samples p[–1, y] and y = 0..3 are marked as “available for Intra chroma prediction”,
- 如果象素 p[x, –1]（x = 0..3）和象素 p[–1, y] （y = 0..3）都被标识为“可用作帧内色差预测”，那么

$$pred_C[x, y] = \left(\sum_{x'=0}^{3} p[x', -1] + \sum_{y'=0}^{3} p[-1, y'] + 4 \right) >> 3, \text{ with } x = 0..3 \text{ and } y = 0..3 \quad (8\text{-}82)$$

- If the samples p[x, –1] with x = 0..3 are marked as “available for Intra chroma prediction” and the samples p[–1, y] with y = 0..3 are marked as “not available for Intra chroma prediction”,
- 如果象素 p[x, –1]（x = 0..3）被标识为“可用作帧内色差预测”，而象素 p[–1, y] 被标识为“不可用作帧内色差预测”，那么

$$pred_C[x, y] = \left(\sum_{x'=0}^{3} p[x', -1] + 2 \right) >> 2, \text{ with } x = 0..3 \text{ and } y = 0..3 \quad (8\text{-}83)$$

- If the samples p[x, –1] with x = 0..3 are marked as “not available for Intra chroma prediction” and the samples p[-1, y] with y = 0..3 are marked as “available for Intra chroma prediction”,
- 如果象素 p[x, –1]（x = 0..3）被标识为“不可用作帧内色差预测”，而象素 p[–1, y] 被标识为“可用作帧内色差预测”，那么

$$pred_C[x, y] = \left(\sum_{y'=0}^{3} p[-1, y'] + 2 \right) >> 2, \text{ with } x = 0..3 \text{ and } y = 0..3 \quad (8\text{-}84)$$

- Otherwise (the samples p[x, –1] with x = 0..3 and the samples p[-1, y] with y = 0..3 are marked as “not available for Intra chroma prediction”),
- 否则（象素 p[x, –1]（x = 0..3）和象素 p[–1, y] （y = 0..3）都被标识为“不可用作帧内色差预测），

$$pred_C[x, y] = 128, \text{ with } x = 0..3 \text{ and } y = 0..3 \quad (8\text{-}85)$$

The values of the prediction samples $pred_C[x, y]$ with x = 4..7 and y = 0..3 are derived as follows.

预测象素 $pred_C[x, y]$（x = 4..7和 y = 0..3）的值计算如下：

- If the samples p[x, -1] with x = 4..7 are marked as “available for Intra chroma prediction”,
- 如果象素 p[x, -1]（x = 4..7）被标识为“可用作帧内色差预测”，

$$pred_C[x, y] = \left(\sum_{x'=4}^{7} p[x', -1] + 2 \right) >> 2, \text{ with } x = 4..7 \text{ and } y = 0..3 \quad (8\text{-}86)$$

- If the samples p[-1, y] with y = 0..3 are marked as “available for Intra chroma prediction”,
- 如果象素 p[-1, y]（y = 0..3）被标识为“可用作帧内色差预测”，

$$pred_C[x, y] = \left(\sum_{y'=0}^{3} p[-1, y'] + 2 \right) >> 2, \text{ with } x = 4..7 \text{ and } y = 0..3 \quad (8\text{-}87)$$

- Otherwise (the samples p[x, –1] with x = 4..7 and the samples p[-1, y] with y = 0..3 are marked as “not available for Intra chroma prediction”),
- 否则（象素p[x, –1]（x = 4..7）和象素p[-1, y]（y = 0..3）都被标识为“不可用作帧内色差预测”），

$$\text{pred}_C[\,x, y\,] = 128, \text{ with } x = 4..7 \text{ and } y = 0..3 \qquad (8\text{-}88)$$

The values of the prediction samples $\text{pred}_C[\,x, y\,]$ with x = 0..3 and y = 4..7 are derived as follows.

预测象素 $\text{pred}_C[\,x, y\,]$（x = 0..3 和 y = 4..7）的值计算如下：

- If the samples p[-1, y] with y = 4..7 are marked as “available for Intra chroma prediction”,
- 如果象素p[-1, y] （ y = 4..7）被标识为“可用作帧内色差预测”，

$$\text{pred}_C[\,x, y\,] = \left(\sum_{y'=4}^{7} p[-1, y'] + 2 \right) >> 2, \text{ with } x = 0..3 \text{ and } y = 4..7 \qquad (8\text{-}89)$$

- If the samples p[x, -1] with x = 0..3 are marked as “available for Intra chroma prediction”,
- 如果象素p[x, -1] （ x = 0..3）被标识为“可用作帧内色差预测”，

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=0}^{3} p[x', -1] + 2 \right) >> 2, \text{ with } x = 0..3 \text{ and } y = 4..7 \qquad (8\text{-}90)$$

- Otherwise (the samples p[x, –1] with x = 0..3 and the samples p[-1, y] with y = 4..7 are marked as “not available for Intra chroma prediction”),
- 否则（象素p[x, –1] （ x = 0..3）和 象素p[-1, y]（ y = 4..7）都被标识为“不可用作帧内色差预测”），

$$\text{pred}_C[\,x, y\,] = 128, \text{ with } x = 0..3 \text{ and } y = 4..7 \qquad (8\text{-}91)$$

The values of the prediction samples $\text{pred}_C[\,x, y\,]$ with x = 4..7 and y = 4..7 are derived as follows.

预测象素 $\text{pred}_C[\,x, y\,]$（x = 4..7 和 y = 4..7）的值计算如下：

- If the samples p[x, –1] with x = 4..7 and the samples p[–1, y] and y = 4..7 are marked as “available for Intra chroma prediction”,
- 如果象素p[x, –1] （ x = 4..7 ）和象素 p[–1, y] （ y = 4..7）都被标识为“可用作帧内色差预测”，

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=4}^{7} p[x', -1] + \sum_{y'=4}^{7} p[-1, y'] + 4 \right) >> 3, \text{ with } x = 4..7 \text{ and } y = 4..7 \qquad (8\text{-}92)$$

- If the samples p[x, –1] with x = 4..7 are marked as “available for Intra chroma prediction” and the samples p[–1, y] with y = 4..7 are marked as “not available for Intra chroma prediction”,
- 如果象素 p[x, –1] （x = 4..7）被标识为“可用作帧内色差预测”而象素p[–1, y] （ y = 4..7）被标识为“不可用作帧内色差预测”，

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=4}^{7} p[x', -1] + 2 \right) >> 2, \text{ with } x = 4..7 \text{ and } y = 4..7 \qquad (8\text{-}93)$$

- If the samples p[x, –1] with x = 4..7 are marked as “not available for Intra chroma prediction” and the samples p[-1, y] with y = 4..7 are marked as “available for Intra chroma prediction”,
- 如果象素 p[x, –1] （x = 4..7）被标识为“不可用作帧内色差预测”而象素p[–1, y] （ y = 4..7）被标识为“可用作帧内色差预测”，

$$\text{pred}_C[\,x, y\,] = \left(\sum_{y'=4}^{7} p[-1, y'] + 2 \right) >> 2, \text{ with } x = 4..7 \text{ and } y = 4..7 \qquad (8\text{-}94)$$

- Otherwise (the samples p[x, –1] with x = 4..7 and the samples p[-1, y] with y = 4..7 are marked as “not available for Intra chroma prediction”),

- 否则（象素p[x, –1] （ x = 4..7 ） 和象素 p[–1, y] （ y = 4..7）都被标识为“不可用作帧内色差预测”）

$$\mathrm{pred}_C[\,x, y\,] = 128, \text{ with } x = 4..7 \text{ and } y = 4..7 \tag{8-95}$$

8.3.3.2 Specification of Intra_Chroma_Horizontal prediction mode

8.3.3.2 Intra_Chroma_Horizontal 预测模式的说明

This mode shall be used only when the samples p[-1, y] with y = 0..7 are marked as “available for Intra chroma prediction”.

只有当象素 p[-1, y] （y = 0..7）被标识为“可用作帧内色差预测”时才使用这个过程。

The values of the prediction samples pred_C[x, y] are derived as follows.

预测象素pred_C[x, y]的值计算如下：

$$\mathrm{pred}_C[\,x, y\,] = p[\,-1, y\,], \text{ with } x, y = 0..7 \tag{8-96}$$

8.3.3.3 Specification of Intra_Chroma_Vertical prediction mode

8.3.3.3 Intra_Chroma_Vertical 预测模式的说明

This mode shall be used only when the samples p[x, -1] with x = 0..7 are marked as “available for Intra chroma prediction”.

只有当象素 p[x, -1]（x = 0..7）被标识为“可用作帧内色差预测”时才使用这个过程。

The values of the prediction samples pred_C[x, y] are derived as follows.

预测象素pred_C[x, y]的值计算如下：

$$\mathrm{pred}_C[\,x, y\,] = p[\,x, -1\,], \text{ with } x, y = 0..7 \tag{8-97}$$

8.3.3.4 Specification of Intra_Chroma_Plane prediction mode

8.3.3.4 Intra_Chroma_Plane 预测模式的说明

This mode shall be used only when the samples p[x, -1], with x = 0..7 and p[-1, y], with y = -1..7 are marked as “available for Intra chroma prediction”.

只有当象素 p[x, -1]（ x = 0..7）和 p[-1, y]（ y = -1..7）都标识为“可用作帧内色差预测”时才使用这个过程。

The values of the prediction samples pred_C[x, y] are derived as follows.

预测象素pred_C[x, y]的值计算如下：

$$\mathrm{pred}_C[\,x, y\,] = \mathrm{Clip1}(\,(\,a + b * (\,x - 3\,) + c * (\,y - 3\,) + 16\,) >> 5\,), \text{ with } x, y = 0..7 \tag{8-98}$$

where:

这里

$$a = 16 * (\,p[\,-1, 7\,] + p[\,7, -1\,]\,) \tag{8-99}$$

$$b = (\,17 * H + 16\,) >> 5 \tag{8-100}$$

$$c = (\,17 * V + 16\,) >> 5 \tag{8-101}$$

and H and V are specified as follows.

H 和 V 的说明如下：

$$H = \sum_{x'=0}^{3} (x'+1) * \left(p[4 + x', -1] - p[2 - x', -1]\right) \tag{8-102}$$

$$V = \sum_{y'=0}^{3} (y'+1) * \left(p[-1, 4 + y'] - p[-1, 2 - y']\right) \tag{8-103}$$

8.3.4 Sample construction process for I_PCM macroblocks

8.3.4 I_PCM 宏块的象素重建过程

This process is invoked when mb_type is equal to I_PCM.

当宏块的 mb_type 等于 I_PCM时 使用这个过程。

Outputs of this process are constructed macroblock samples S'_L, S'_{Cb}, and S'_{Cr} prior to the deblocking filter process.

本过程的输出是环路滤波之前的重建宏块象素值S'_L, S'_{Cb}, 和 S'_{Cr}。

The variable dy is derived as follows.

变量 dy 的计算如下：

- If MbaffFrameFlag is equal to 1 and the current macroblock is a field macroblock, dy is set equal to 2.
- Otherwise (MbaffFrameFlag is equal to 0 or the current macroblock is a frame macroblock), dy is set equal to 1.
- 如果 MbaffFrameFlag 等于1 并且当前宏块是一场编码宏块，那么 dy 被设置成 2。
- 否则（MbaffFrameFlag 等于0 或者当前宏块是一帧编码宏块），那么 dy 被设置成 1。

The position of the upper-left luma sample of the current macroblock is derived by invoking the inverse macroblock scanning process in subclause 6.4.1 with CurrMbAddr as input and the output being assigned to (xP, yP).

将 CurrMbAddr 作为输入，使用 6.4.1 小节中的反向宏块扫描处理过程，计算出当前宏块的左上角亮度象素的坐标位置，并且赋给(xP, yP)。

The constructed samples prior to the deblocking process are generated as specified by:

环路滤波之前的重建象素值的计算如下：

```
for( i = 0; i < 256; i++ )
    S'L[ xP + ( i % 16 ), yP + dy * ( i / 16 ) ) ] = pcm_byte[ i ]                    (8-104)

for( i = 0; i < 64; i++ ) {
    S'Cb[ ( xP >> 1 ) + ( i % 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm_byte[ i + 256 ]    (8-105)
    S'Cr[ ( xP >> 1 ) + ( i % 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm_byte[ i + 320 ]
}
```

8.4 Inter prediction process

8.4 帧间预测过程

This process is invoked when decoding P and B macroblock types.

当解码 P 或 B 宏块时使用这个过程。

Outputs of this process are Inter prediction samples for the current macroblock that are a 16x16 array $pred_L$ of luma samples and two 8x8 arrays $pred_{Cr}$ and $pred_{Cb}$ of chroma samples, one for each of the chroma components Cb and Cr.

本过程的输出是当前宏块的帧间预测象素值，包括 16x16 的亮度分量和两个 8x8 的色差分量，分别对应Cb和Cr。

The partitioning of a macroblock is specified by mb_type. Each macroblock partition is referred to by mbPartIdx. When the macroblock partitioning consists of partitions that are equal to sub-macroblocks, each sub-macroblock can be further partitioned into sub-macroblock partitions as specified by sub_mb_type. Each sub-macroblock partition is referred to by subMbPartIdx. When the macroblock partitioning does not consist of sub-macroblocks, subMbPartIdx is set equal to 0.

宏块的分割方式由 mb_type 来表示，分割后的每一个宏块部分（macroblock partition）由 mbPartIdx 来表示。当每一个宏块部分都是一个子宏块时（sub-macroblocks），那么每一个子宏块可以被进一步的划分成子宏块部分（sub-macroblock partitions），分割方式 sub_mb_type 来表示，每个子宏块部分由 subMbPartIdx 来表示。当宏块部分不都是子宏块时，那么 subMbPartIdx 将被设置成 0 。

The following steps are specified for each macroblock partition or for each sub-macroblock partition.

下面的步骤说明了分割宏块或分割子宏块的过程。

The functions MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ), and SubMbPartHeight( ) describing the width and height of macroblock partitions and sub-macroblock partitions are specified in Table 7-10, Table 7-11, Table 7-14, and Table 7-15.

函数 MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ), 和 SubMbPartHeight( ) 描述了宏块部分或者子宏块部分的宽度和高度，如 Table 7-10, Table 7-11, Table 7-14, and Table 7-15 所示。

The variables partWidth and partHeight are derived as follows.

变量 partWidth 和 partHeight 的计算方法如下：

- If mb_type is not equal to P_8x8 or P_8x8ref0 or B_8x8, the following applies.
- 如果 mb_type 不等于 P_8x8 或者 P_8x8ref0 或者 B_8x8，那么使用下面的过程：

partWidth = MbPartWidth(mb_type) (8-106)

partHeight = MbPartHeight(mb_type) (8-107)

- Otherwise (mb_type is equal to P_8x8 or P_8x8ref0 or B_8x8),
- 否则（mb_type等于 P_8x8 或者 P_8x8ref0 或者 B_8x8）

partWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]) (8-108)

partHeight = SubMbPartHeight(sub_mb_type[mbPartIdx]). (8-109)

When mb_type is equal to B_Skip or B_Direct_16x16 or sub_mb_type[mbPartIdx] is equal to B_Direct_8x8, the Inter prediction process is specified for

当 mb_type 等于 B_Skip 或者 B_Direct_16x16 或者 sub_mb_type[mbPartIdx] 等于 B_Direct_8x8 时，帧间预测过程如下：

partWidth = 4 (8-110)

partHeight = 4 (8-111)

with mbPartIdx proceeding over values 0..3 and for each sub-macroblock indexed by mbPartIdx, subMbPartIdx proceeds over values 0..3.

mbPartIdx 的范围是从 0 到 3，对于每一个索引为 mbPartIdx 的子宏块 sub-macroblock 来说，subMbPartIdx 的范围是从 0 到 3。

The Inter prediction process for a macroblock partition mbPartIdx and a sub-macroblock partition subMbPartIdx consists of the following ordered steps

对于宏块部分mbPartIdx 和子宏块部分subMbPartIdx，它们的帧间预测步骤如下：

1. Derivation process for motion vector components and reference indices as specified in subclause 8.4.1.

   运动矢量分量和参考索引的计算，如 8.4.1 所示。

   Inputs to this process are

   本过程的输入是：

   - a macroblock partition mbPartIdx,
   - a sub-macroblock partition subMbPartIdx.
   - 宏块部分的索引 mbPartIdx，
   - 子宏块部分的索引 subMbPartIdx。

   Outputs of this process are

   本过程的输出是：

   - luma motion vectors mvL0 and mvL1 and the chroma motion vectors mvCL0 and mvCL1
   - reference indices refIdxL0 and refIdxL1

- prediction list utilization flags predFlagL0 and predFlagL1
- 亮度分量的运动矢量 mvL0 和 mvL1 和 色差分量的运动矢量 mvCL0 和 mvCL1
- 参考索引 refIdxL0 和 refIdxL1
- 预测列表使用标志 predFlagL0 和 predFlagL1

2. Decoding process for Inter prediction samples as specified in subclause 8.4.2.

   帧间预测象素的解码过程，如 8.4.2 所示。

   Inputs to this process are

   本过程的输入是：

- a macroblock partition mbPartIdx,
- a sub-macroblock partition subMbPartIdx.
- variables specifying partition width and height, partWidth, and partHeight
- luma motion vectors mvL0 and mvL1 and the chroma motion vectors mvCL0 and mvCL1
- reference indices refIdxL0 and refIdxL1
- prediction list utilization flags predFlagL0 and predFlagL1
- 宏块部分的索引 mbPartIdx
- 子宏块部分的索引 subMbPartIdx
- mbPartIdx或 subMbPartIdx 的宽度和高度变量 partWidth 和 partHeight
- 亮度分量的运动矢量 mvL0 和 mvL1 和 色差分量的运动矢量 mvCL0 和 mvCL1
- 参考索引 refIdxL0 和 refIdxL1
- 预测列表使用标志 predFlagL0 和 predFlagL1

   Outputs of this process are

   本过程的输出是：

- inter prediction samples (pred); which are a (partWidth)x(partHeight) array $predPart_L$ of prediction luma samples and two (partWidth/2)x(partHeight/2) arrays $predPart_{Cr}$, and $predPart_{Cb}$ of prediction chroma samples, one for each of the chroma components Cb and Cr.
- 帧间预测象素值（pred），它们是一个 (partWidth)x(partHeight)预测亮度象素数组 $predPart_L$和两个(partWidth/2)x(partHeight/2) 预测色差象素数组 $predPart_{Cr}$, 和 $predPart_{Cb}$ 分别对应于色差分量 Cb和 Cr。

For use in derivation processes of variables invoked later in the decoding process, the following assignments are made:

为了以后解码过程中变量的使用，进行如下的赋值过程：

MvL0[mbPartIdx][subMbPartIdx] = mvL0 (8-112)

MvL1[mbPartIdx][subMbPartIdx] = mvL1 (8-113)

RefIdxL0[mbPartIdx] = refIdxL0 (8-114)

RefIdxL1[mbPartIdx] = refIdxL1 (8-115)

PredFlagL0[mbPartIdx] = predFlagL0 (8-116)

PredFlagL1[mbPartIdx] = predFlagL1 (8-117)

The location of the upper-left sample of the partition relative to the upper-left sample of the macroblock is derived by invoking the inverse macroblock partition scanning process as described in subclause 6.4.2.1 with mbPartIdx as the input and (xP, yP) as the output.

将 mbPartIdx 作为输入，使用 6.4.2.1 描述的过程计算出当前部分（partition）的左上角象素相对于当前宏块的左上角象素的坐标位置(xP, yP)并且输出。

The location of the upper-left sample of the macroblock sub-partition relative to the upper-left sample of the macroblock partition is derived by invoking the inverse sub-macroblock partition scanning process as described in subclause 6.4.2.2 with subMbPartIdx as the input and (xS, yS) as the output.

将 subMbPartIdx 作为输入，使用 6.4.2.2 描述的过程计算出当前子宏块部分（sub-partition）的左上角象素相对于当前子宏块的左上角象素的坐标位置 (xS, yS) 并且作为输出。

The macroblock prediction is formed by placing the partition or sub-macroblock partition prediction samples in their correct relative positions in the macroblock, as follows.

将宏块部分或者子宏块部分的预测象素值放在当前宏块中对应的位置从而形成当前宏块的预测值。

The variable $pred_L[\, xP + xS + x, yP + yS + y \,]$ with x = 0 .. partWidth – 1, y = 0 .. partHeight – 1 is derived by

变量 $pred_L[\, xP + xS + x, yP + yS + y \,]$ （ x = 0 .. partWidth – 1, y = 0 .. partHeight – 1）的计算如下：

$$pred_L[\, xP + xS + x, yP + yS + y \,] = predPart_L[\, x, y \,] \tag{8-118}$$

The variable $pred_C[\, xP / 2 + xS / 2 + x, yP / 2 + yS / 2 + y \,]$ with x = 0 .. partWidth/2 – 1, y = 0 .. partHeight/2 – 1, and C being replaced by Cb or Cr is derived by

变量 $pred_C[\, xP / 2 + xS / 2 + x, yP / 2 + yS / 2 + y \,]$（ x = 0 .. partWidth/2 – 1, y = 0 .. partHeight/2 – 1）的计算如下（这里C 可能是 Cb 或者 Cr ）：

$$pred_C[\, xP / 2 + xS / 2 + x, yP / 2 + yS / 2 + y \,] = predPart_C[\, x, y \,] \tag{8-119}$$

8.4.1 Derivation process for motion vector components and reference indices

8.4.1 运动矢量和参考索引的计算过程

Inputs to this process are

本过程的输入是：

- a macroblock partition mbPartIdx,
- a sub-macroblock partition subMbPartIdx.
- 宏块部分的索引 mbPartIdx，
- 子宏块部分的索引 subMbPartIdx。

Outputs of this process are

本过程的输出是：

- luma motion vectors mvL0 and mvL1 as well as the chroma motion vectors mvCL0 and mvCL1
- reference indices refIdxL0 and refIdxL1
- prediction list utilization flags predFlagL0 and predFlagL1
- 亮度分量的运动矢量 mvL0 和 mvL1 和 色差分量的运动矢量 mvCL0 和 mvCL1
- 参考索引 refIdxL0 和 refIdxL1
- 预测列表使用标志 predFlagL0 和 predFlagL1

For the derivation of the variables mvL0 and mvL1 as well as refIdxL0 and refIdxL1, the following applies.

为了计算变量 mvL0 、 mvL1 和 refIdxL0 、 refIdxL1，使用下面的过程：

- If mb_type is equal to P_Skip, the derivation process for luma motion vectors for skipped macroblocks in P and SP slices in subclause 8.4.1.1 is invoked with the output being the luma motion vectors mvL0 and reference indices refIdxL0, and predFlagL0 is set equal to 1. mvL1 and refIdxL1 are marked as not available and predFlagL1 is set equal to 0.

- 如果 mb_type 等于 P_Skip，那么在 P 和 SP slice 中亮度分量运动矢量的计算使用 8.4.1.1 小节中的方法，输出为亮度分量的运动矢量 mvL0 和参考索引 refIdxL0，predFlagL0设置成 1，而 mvL1 和 refIdxL1 设置成不可用，predFlagL0设置成 0。

- If mb_type is equal to B_Skip, or B_Direct_16x16 or sub_mb_type[subMbPartIdx] is equal to B_Direct_8x8, the derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8 in B slices in subclause 8.4.1.2 is invoked with mbPartIdx and subMbPartIdx as the input and the output being the luma motion vectors mvL0, mvL1, the reference indices refIdxL0, refIdxL1, and the prediction utilization flags predFlagL0 and predFlagL1.

- 如果 mb_type 等于 B_Skip, 或 B_Direct_16x16 或者 sub_mb_type[subMbPartIdx] 等于 B_Direct_8x8,，那么将 mbPartIdx 和 subMbPartIdx 作为输入，使用 8.4.1.2 小节中的方法计算出 B slice 中B_Skip, B_Direct_16x16, 和 B_Direct_8x8 的亮度分量运动矢量mvL0, mvL1、参考索引refIdxL0, refIdxL1和预测使用标志 predFlagL0 和 predFlagL1并输出。

- Otherwise, for X being replaced by either 0 or 1 in the variables predFlagLX, mvLX, refIdxLX, and in Pred_LX and in the syntax elements ref_idx_lX and mvd_lX, the following is specified.

- 否则，使用下面的过程，变量 predFlagLX, mvLX, refIdxLX, Pred_LX和语法元素 ref_idx_lX 和 mvd_lX 中的X 可能是 0 或者 1。

  - If MbPartPredMode(mb_type, mbPartIdx) is equal to Pred_LX or to BiPred,

  - 如果 MbPartPredMode(mb_type, mbPartIdx) 等于 Pred_LX 或者 BiPred

    refIdxLX = ref_idx_lX[mbPartIdx] (8-120)

    predFlagLX = 1 (8-121)

  - Otherwise

  - 否则

    refIdxLX = -1 (8-122)

    predFlagLX = 0 (8-123)

  and the derivation process for luma motion vector prediction in subclause 8.4.1.3 is invoked with mbPartIdx subMbPartIdx, refIdxLX, and list suffix LX as the input and the output being mvpLX. The luma motion vectors are derived as follows.

  并且将 mbPartIdx，subMbPartIdx, refIdxLX, 和 LX作为输入，使用 8.4.1.3 小节中的亮度运动矢量预测的处理过程计算出 mvpLX，亮度运动矢量的计算过程如下：

    mvLX[0] = mvpLX[0] + mvd_lX[mbPartIdx][subMbPartIdx][0] (8-124)

    mvLX[1] = mvpLX[1] + mvd_lX[mbPartIdx][subMbPartIdx][1] (8-125)

For the derivation of the variables for the chroma motion vectors, the following applies. When predFlagLX (with X being either 0 or 1) is equal to 1, the derivation process for chroma motion vectors in subclause 8.4.1.4 is invoked with mvLX and refIdxLX as input and the output being mvCLX.

对于色差运动矢量变量的计算，使用下面的过程。当 predFlagLX （X 可能是 0 或 1）等于 1 时，那么将 mvLX 和 refIdxLX 作为输入，使用 8.4.1.4 小节中的色差运动矢量的处理过程计算出 mvCLX 并输出。

8.4.1.1 Derivation process for luma motion vectors for skipped macroblocks in P and SP slices

8.4.1.1 在 P 和 SP slice 中跳过宏块的运动矢量的处理过程

This process is invoked when mb_type is equal to P_Skip.

当 mb_type 等于 P_Skip 时使用这个处理过程。

Outputs of this process are the motion vector mvL0 and the reference index refIdxL0.

本过程的输出是运动矢量 mvL0 和参考索引 refIdxL0。

The reference index refIdxL0 for a skipped macroblock is derived as follows.

对于跳过宏块来说，参考索引 refIdxL0 的计算如下：

$$\text{refIdxL0} = 0. \tag{8-126}$$

For the derivation of the motion vector mvL0 of a P_Skip macroblock type, the following applies.

对于 P_Skip 宏块运动矢量的计算，使用下面的过程。

- The process specified in subclause 8.4.1.3.2 is invoked with mbPartIdx set equal to 0, subMbPartIdx set equal to 0, and list suffix L0 as input and the output is assigned to mbAddrA, mbAddrB, mvL0A, mvL0B, refIdxL0A, and refIdxL0B.
- 将设置成 0 的 mbPartIdx 、subMbPartIdx 和 列表后缀 L0 作为输入，使用 8.4.1.3.2 小节中的处理过程计算出 mbAddrA, mbAddrB, mvL0A, mvL0B, refIdxL0A, 和 refIdxL0B 并输出。
- If any one of the following conditions is true, both components of the motion vector mvL0 are set equal to 0.
- 如果下面的任何一个条件成立，那么运动矢量 mvL0 的两个分量都被设置成 0。
    - mbAddrA is not available
    - mbAddrB is not available
    - refIdxL0A is equal to 0 and both components of mvL0A are equal to 0
    - refIdxL0B is equal to 0 and both components of mvL0B are equal to 0
    - mbAddrA 不可用
    - mbAddrB 不可用
    - refIdxL0A 等于 0 并且 mvL0A 的两个分量都为 0
    - refIdxL0B 等于 0 并且 mvL0B 的两个分量都为 0
- Otherwise, the derivation process for luma motion vector prediction as specified in subclause 8.4.1.3 is invoked with mbPartIdx = 0, subMbPartIdx = 0, refIdxL0, and list suffix L0 as input and the output is assigned to mvL0.
- 否则，将mbPartIdx = 0, subMbPartIdx = 0, refIdxL0, 和列表后缀 L0作为输入，使用 8.4.1.3 小节中的亮度分量运动矢量预测的处理过程并输出 mvL0。

NOTE – The output is directly assigned to mvL0, since the predictor is equal to the actual motion vector.
注 — 输出值直接赋给 mvL0，因为预测值就等于实际的运动矢量。

8.4.1.2 Derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8

8.4.1.2 对于 B_Skip, B_Direct_16x16, 和 B_Direct_8x8的亮度运动矢量的处理过程

This process is invoked when mb_type is equal to B_Skip or B_Direct_16x16, or sub_mb_type[mbPartIdx] is equal to B_Direct_8x8.

当 mb_type 等于 B_Skip 、 B_Direct_16x16 或者 sub_mb_type[mbPartIdx] 等于 B_Direct_8x8 时使用本处理过程。

Inputs to this process are mbPartIdx and subMbPartIdx.

本过程的输入是 mbPartIdx 和 subMbPartIdx。

Outputs of this process are the reference indices refIdxL0, refIdxL1, the motion vectors mvL0 and mvL1, and the prediction list utilization flags, predFlagL0 and predFlagL1.
本过程的输出是参考索引 refIdxL0和refIdxL1，运动矢量 mvL0 和 mvL1，以及预测列表使用标志 predFlagL0 和 predFlagL1。

The derivation process depends on the value of direct_spatial_mv_pred_flag in the slice header syntax as specified in subclause 7.3.3.
本处理过程依懒于 slice 头语法中的 direct_spatial_mv_pred_flag 值，该变量的定义如 7.3.3 小节所示。

- If direct_spatial_mv_pred_flag is equal to 1, the mode in which the outputs of this process are derived is referred to as spatial direct prediction mode.

- Otherwise (direct_spatial_mv_pred_flag is equal to 0), mode in which the outputs of this process are derived is referred to as temporal direct prediction mode.
- 如果 direct_spatial_mv_pred_flag 等于 1，那么本过程的输出模式被称为空间直接预测模式。
- 否则（direct_spatial_mv_pred_flag 等于 0），本过程的输出模式被称为时间直接预测模式。

Both spatial and temporal direct prediction mode use the co-located motion vectors and reference indices as specified in subclause 8.4.1.2.1.

空间和时间直接预测模式都使用对应位置（co-located）的运动矢量和参考索引，如 8.4.1.2.1 小节所示。

The motion vectors and reference indices are derived as follows.

运动矢量和参考索引的计算方法如下：

- If direct_spatial_mv_pred_flag is equal to 0, the temporal direct motion vector and reference index prediction mode specified in subclause 8.4.1.2.3 is used.
- Otherwise (direct_spatial_mv_pred_flag is equal to 1), the spatial direct motion vector and reference index prediction mode specified in subclause 8.4.1.2.2 is used.
- 如果 direct_spatial_mv_pred_flag 等于 0，那么使用 8.4.1.2.3 小节中的时间直接运动矢量和参考索引的预测模式。
- 否则（direct_spatial_mv_pred_flag 等于 1），使用 8.4.1.2.2 小节中的空间直接运动矢量和参考索引的预测模式。

8.4.1.2.1 Derivation process for the co-located 4x4 sub-macroblock partitions

8.4.1.2.1 对应位置 4x4 子宏块部分的处理过程

Inputs to this process are mbPartIdx and subMbPartIdx.

本过程的输入是 mbPartIdx 和 subMbPartIdx。

Outputs of this process are the picture colPic, the co-located macroblock mbAddrCol, the motion vector mvCol, the reference index refIdxCol, and the variable vertMvScale (which can be One_To_One, Frm_To_Fld or Fld_To_Frm).

本过程的输出是图像 colPic，对应位置的宏块 mbAddrCol，运动矢量 mvCol，参考索引 refIdxCol 和 变量 vertMvScale（它可能是One_To_One, Frm_To_Fld 或 Fld_To_Frm）。

Let firstRefPicL1 be the reference picture referred by RefPicList1[0].

假设 firstRefPicL1 为参考图像 RefPicList1[0]。

When firstRefPicL1 is a frame or a complementary field pair, let firstRefPicL1Top and firstRefPicL1Bottom be the top and bottom fields of firstRefPicL1, and let the following variables be specified as

当 firstRefPicL1 为一帧图像或一互补场对，那么假设 firstRefPicL1Top 和 firstRefPicL1Bottom 分别为 firstRefPicL1的顶场和底场，并且下面的变量说明如下：

$$\text{topAbsDiffPOC} = \text{Abs}(\ \text{DiffPicOrderCnt}(\ \text{firstRefPicL1Top}, \text{CurrPic}\ )\ ) \tag{8-127}$$

$$\text{bottomAbsDiffPOC} = \text{Abs}(\ \text{DiffPicOrderCnt}(\ \text{firstRefPicL1Bottom}, \text{CurrPic}\ )\ ) \tag{8-128}$$

The variable colPic specifies the picture that contains the co-located macroblock as specified in Table 8-5.

变量 colPic 表示了包含对应位置宏块的图像，如表 Table 8-5 所示。

Table 8-5 – Specification of the variable colPic

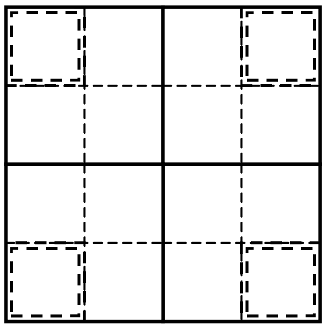
field_pic_flag	The first entry in RefPicList1 is …	mb_field_decoding_flag	additional condition	colPic
1	a field of a decoded frame			the frame containing firstRefPicL1
	a decoded field			firstRefPicL1
0	a decoded frame			firstRefPicL1

	a complementary field pair	0	topAbsDiffPOC < bottomAbsDiffPOC	the top field of firstRefPicL1
			topAbsDiffPOC >= bottomAbsDiffPOC	the bottom field of firstRefPicL1
		1	(CurrMbAddr & 1) = = 0	the top field of firstRefPicL1
			(CurrMbAddr & 1) != 0	the bottom field of firstRefPicL1

When direct_8x8_inference_flag is equal to 1, the motion vectors associated with the outside macroblock corner 4x4 sub-macroblock partition are used for all 4x4 sub-macroblock partitions of a macroblock partition. In this case, subMbPartIdx is set equal to

（这里的理解有问题）

当 direct_8x8_inference_flag 等于 1 时， 将和外面宏块角落 4x4 子宏块部分相关的运动矢量应用于一宏块部分的所有 4x4 子宏块部分。在这种情况下，subMbPartIdx的设置如下：（参考JVT-F025和JVT-G037）



subMbPartIdx = mbPartIdx (8-129)

Let PicCodingStruct(X) be a function with the argument X being either CurrPic or colPic. It is specified in Table 8-6.

假设PicCodingStruct(X)为 X 的函数，X 可能为 CurrPic 或者 colPic，具体说明如表 Table 8-6。

Table 8-6 – Specification of PicCodingStruct(X)

X is coded with field_pic_flag equal to …	mb_adaptive_frame_field_flag	PicCodingStruct(X)
1		FLD
0	0	FRM
0	1	AFRM

With luma4x4BlkIdx = mbPartIdx * 4 + subMbPartIdx, the inverse 4x4 luma block scanning process as specified in subclause 6.4.3 is invoked with luma4x4BlkIdx as the input and (x, y) assigned to (xCol, yCol) as the output.

假设 luma4x4BlkIdx = mbPartIdx * 4 + subMbPartIdx，将 luma4x4BlkIdx 作为输入，使用 6.4.3 小节中的反向 4x4 亮度块扫描过程计算出（x，y），赋给(xCol, yCol)并输出。

Table 8-7 specifies the co-located macroblock address mbAddrCol, yM, and the variable vertMvScale in two steps:

表 Table 8-7 分两步说明了对应宏块的地址 mbAddrCol, yM, 和变量 vertMvScale。

1. Specification of a macroblock address mbAddrX depending on PicCodingStruct(CurrPic), and PicCodingStruct(colPic).

   NOTE - It is not possible for CurrPic and colPic picture coding types to be either (FRM, AFRM) or (AFRM, FRM) because these picture coding types must be separated by an IDR picture.

2. Specification of mbAddrCol, yM, and vertMvScale depending on mb_field_decoding_flag and the following variable

   - If the macroblock mbAddrX in the picture colPic is a field macroblock, fieldDecodingFlagX is set equal to 1
   - Otherwise (the macroblock mbAddrX in the picture colPic is a frame macroblock), fieldDecodingFlagX is set equal to 0.

1．与 PicCodingStruct(CurrPic)相关的宏块地址 mbAddrX 的表示，和 PicCodingStruct(colPic)。

注 － 对于 CurrPic 和 colPic 图像编码类型来说，不可能出现 (FRM, AFRM) 或者 (AFRM, FRM)，因为这些图像编码类型必须用 IDR 隔开。

2．与 mb_field_decoding_fla相关的 mbAddrCol, yM, vertMvScale 的表示，和下面的变量

- 在 colPic 图像中地址为 mbAddrX 的宏块是一场宏块，那么 fieldDecodingFlagX被设置成 1。
- 否则（在 colPic 图像中地址为 mbAddrX 的宏块是一帧宏块），那么 fieldDecodingFlagX被设置成 0。

Unspecified values in Table 8-7 indicate that the value of the corresponding variable is not relevant for the current table row.

在 Table 8-7 中没有说明的值表示对应变量的值和表中对应的行无关。

mbAddrCol is set equal to CurrMbAddr or to one of the following values.

MbAddrCol 被设置成 CurrMbAddr 或者下面的一个值。

$$\begin{aligned} \text{mbAddrCol1} = {} & 2 * \text{PicWidthInMbs} * (\text{CurrMbAddr} / \text{PicWidthInMbs}) + \\ & (\text{CurrMbAddr} \,\%\, \text{PicWidthInMbs}) + \text{PicWidthInMbs} * (\text{yCol} / 8) \end{aligned} \tag{8-130}$$

$$\text{mbAddrCol2} = 2 * \text{CurrMbAddr} + (\text{yCol} / 8) \tag{8-131}$$

$$\text{mbAddrCol3} = 2 * \text{CurrMbAddr} + \text{bottom\_field\_flag} \tag{8-132}$$

$$\begin{aligned} \text{mbAddrCol4} = {} & \text{PicWidthInMbs} * (\text{CurrMbAddr} / (2 * \text{PicWidthInMbs})) + \\ & (\text{CurrMbAddr} \,\%\, \text{PicWidthInMbs}) \end{aligned} \tag{8-133}$$

$$\text{mbAddrCol5} = \text{CurrMbAddr} / 2 \tag{8-134}$$

$$\text{mbAddrCol6} = 2 * (\text{CurrMbAddr} / 2) + ((\text{topAbsDiffPOC} < \text{bottomAbsDiffPOC}) \; ? \; 0 : 1) \tag{8-135}$$

$$\text{mbAddrCol7} = 2 * (\text{CurrMbAddr} / 2) + (\text{yCol} / 8) \tag{8-136}$$

Table 8-7 – Specification of mbAddrCol, yM, and vertMvScale

PicCodingStruct(CurrPic)	PicCodingStruct(colPic)	mbAddrX	mb_field_decoding_flag	fieldDecodingFlagX	mbAddrCol	yM	vertMvScale
FLD	FLD				CurrMbAddr	yCol	One_To_One
	FRM				mbAddrCol1	(2 * yCol) % 16	Frm_To_Fld
	AFRM	2*CurrMbAddr		0	mbAddrCol2	(2 * yCol) % 16	Frm_To_Fld
				1	mbAddrCol3	yCol	One_To_One
FRM	FLD				mbAddrCol4	8 * ((CurrMbAddr / PicWidthInMbs) % 2) + 4 * (yCol / 8)	Fld_To_Frm
	FRM				CurrMbAddr	yCol	One_To_One
AFRM	FLD		0		mbAddrCol5	8 * (CurrMbAddr % 2) +4 * (yCol / 8)	Fld_To_Frm
			1		mbAddrCol5	yCol	One_To_One
	AFRM	CurrMbAddr	0	0	CurrMbAddr	yCol	One_To_One
				1	mbAddrCol6	8 * (CurrMbAddr % 2) + 4 * (yCol / 8)	Fld_To_Frm
		CurrMbAddr	1	0	mbAddrCol7	(2 * yCol) % 16	Frm_To_Fld
				1	CurrMbAddr	yCol	One_To_One

Let mbPartIdxCol be the macroblock partition index of the co-located partition and subMbPartIdxCol the sub-macroblock partition index of the co-located sub-macroblock partition. The partition in the macroblock mbAddrCol inside the picture colPic covering the sample (xCol, yM) shall be assigned to mbPartIdxCol and the sub-macroblock partition inside the partition mbPartIdxCol covering the sample (xCol, yM) in the macroblock mbAddrCol inside the picture colPic shall be assigned to subMbPartIdxCol.

假设 mbPartIdxCol 为对应位置部分的宏块部分索引，subMbPartIdxCol 为对应位置子宏块部分的子宏块部分索引。那么在图像 colPic 中包含象素 (xCol, yM)的宏块部分被分配给mbPartIdxCol，在地址为mbAddrCol的宏块内包含象素 (xCol, yM) 的子宏块部分被分配给 subMbPartIdxCol。

The prediction utilization flags predFlagL0Col and predFlagL1Col are set equal to PredFlagL0[mbPartIdxCol] and PredFlagL1[mbPartIdxCol], respectively, which are the prediction utilization flags that have been assigned to the macroblock partition mbAddrCol\mbPartIdxCol inside the picture colPic.

预测使用标志 predFlagL0Col 和 predFlagL1Col 被分别设置成 PredFlagL0[mbPartIdxCol] 和 PredFlagL1[mbPartIdxCol]，它们在图像 colPic 中已经被分配给宏块部分 mbAddrCol 或者 mbPartIdxCol。

The motion vector mvCol and the reference index refIdxCol are derived as follows.

运动矢量 mvCol 和参考索引 refIdxCol 的计算如下：

- If the macroblock mbAddrCol is coded in Intra macroblock prediction mode or both prediction utilization flags, predFlagL0Col and predFlagL1Col are equal to 0, both components of mvCol are set equal to 0 and refIdxCol is set equal to –1.
- 如果宏块 mbAddrCol 是帧内编码模式或者两个预测使用标志 predFlagL0Col 和 predFlagL1Col 都被设置成 0，那么 mvCol 的两个分量都被设置成 0，refIdxCol 被设置成 –1。
- Otherwise, the following applies.

- 否则，使用下面的过程

  - If predFlagL0Col is equal to 1, the motion vector mvCol and the reference index refIdxCol are set equal to MvL0[mbPartIdxCol][subMbPartIdxCol] and RefIdxL0[mbPartIdxCol], respectively, which are the motion vector mvL0 and the reference index refIdxL0 that have been assigned to the (sub-)macroblock partition mbAddrCol\mbPartIdxCol\subMbPartIdxCol inside the picture colPic.

  - 如果 predFlagL0Col 等于 1，那么运动矢量 mvCol 和参考索引 refIdxCol 将被分别设置成 MvL0[mbPartIdxCol][subMbPartIdxCol] 和 RefIdxL0[mbPartIdxCol]，它们在图像colPic中已经被分配给（子）宏块部分 mbAddrCol\mbPartIdxCol\subMbPartIdxCol 的运动矢量或者参考索引。

  - Otherwise (predFlagL0Col is equal to 0 and predFlagL1Col is equal to 1), the motion vector mvCol and the reference index refIdxCol are set equal to MvL1[mbPartIdxCol][subMbPartIdxCol] and RefIdxL1[mbPartIdxCol], respectively, which are the motion vector mvL1 and the reference index refIdxL1 that have been assigned to the (sub-)macroblock partition mbAddrCol\mbPartIdxCol\subMbPartIdxCol inside the picture colPic.

  - 否则（predFlagL0Col 等于 0， predFlagL1Col 等于 1），运动矢量 mvCol 和参考索引 refIdxCol 将被分别设置成 MvL1[mbPartIdxCol][subMbPartIdxCol] 和 RefIdxL1[mbPartIdxCol]，它们在图像colPic中已经被分配给（子）宏块部分 mbAddrCol\mbPartIdxCol\subMbPartIdxCol 的运动矢量或者参考索引。

8.4.1.2.2 Derivation process for spatial direct luma motion vector and reference index prediction mode

8.4.1.2.2 空间直接亮度运动矢量和参考索引预测模式的处理过程

This process is invoked when direct_spatial_mv_pred_flag is equal to 1 and any of the following conditions is true.

当 direct_spatial_mv_pred_flag 等于 1 并且下面的任何一个条件满足时使用这个过程。

- mb_type is equal to B_Skip
- mb_type is equal to B_Direct_16x16
- sub_mb_type[mbPartIdx] is equal to B_Direct_8x8.
- mb_type 等于 B_Skip
- B_Skip 等于B_Direct_16x16
- sub_mb_type[mbPartIdx] 等于 B_Direct_8x8

Inputs to this process are mbPartIdx, subMbPartIdx.

本过程的输入是mbPartIdx, subMbPartIdx

Outputs of this process are the reference indices refIdxL0, refIdxL1, the motion vectors mvL0 and mvL1, and the prediction list utilization flags, predFlagL0 and predFlagL1.

本过程的输出是参考索引 refIdxL0、refIdxL1,运动矢量mvL0 、mvL1,和预测列表使用标志 predFlagL0 和 predFlagL1

The reference indices refIdxL0 and refIdxL1 and the variable directZeroPredictionFlag are derived by applying the following ordered steps.

参考索引 refIdxL0 和 refIdxL1 以及变量 directZeroPredictionFlag 的计算主要有以下几个步骤：

1. The process specified in subclause 8.4.1.3.2 is invoked with mbPartIdx = 0, subMbPartIdx = 0, and list suffix L0 as input and the output is assigned to the motion vectors mvL0N and the reference indices refIdxL0N with N being replaced by A, B, or C.

   将 mbPartIdx = 0, subMbPartIdx = 0, 和列表后缀 L0 作为输入，使用 8.4.1.3.2 小节中的处理过程，输出赋给运动矢量 mvL0N 和 参考索引 refIdxL0N，这里 N 可能是 A、B 或者 C。

2. The process specified in subclause 8.4.1.3.2 is invoked with mbPartIdx = 0, subMbPartIdx = 0, and list suffix L1 as input and the output is assigned to the motion vectors mvL1N and the reference indices refIdxL1N with N being replaced by A, B, or C.

   将 mbPartIdx = 0, subMbPartIdx = 0, 和列表后缀 L1 作为输入，使用 8.4.1.3.2 小节中的处理过程，输出赋给运动矢量 mvL1N 和 参考索引 refIdxL1N，这里 N 可能是 A、B 或者 C。

NOTE – The motion vectors mvL0N, mvL1N and the reference indices refIdxL0N, refIdxL1N are identical for all 4x4 sub-macroblock partitions of a macroblock.

注 － 对于一个宏块的所有 4x4子宏块来说，运动矢量 mvL0N, mvL1N 和参考索引 refIdxL0N, refIdxL1N 都是相同的。（如果 sub_mb_type[mbPartIdx] 等于 B_Direct_8x8 ，那么应该是子宏块的所有 4x4子宏块）

3. The reference indices refIdxL0, refIdxL1, and directZeroPredictionFlag are derived by

参考帧索引refIdxL0, refIdxL1, 和 directZeroPredictionFlag 的计算如下：

$$refIdxL0 = MinPositive(refIdxL0A, MinPositive(refIdxL0B, refIdxL0C)) \quad (8\text{-}137)$$
$$refIdxL1 = MinPositive(refIdxL1A, MinPositive(refIdxL1B, refIdxL1C)) \quad (8\text{-}138)$$
$$directZeroPredictionFlag = 0 \quad (8\text{-}139)$$

where

这里

$$\mathrm{MinPositive}(x, y) = \begin{cases} \mathrm{Min}(x, y) & \text{if } x >= 0 \text{ and } y >= 0 \\ \mathrm{Max}(x, y) & \text{otherwise} \end{cases} \quad (8\text{-}140)$$

4. When both reference indices refIdxL0 and refIdxL1 are less than 0,

当两个参考索引 refIdxL0 和 refIdxL1 都小于 0 时，

$$refIdxL0 = 0 \quad (8\text{-}141)$$
$$refIdxL1 = 0 \quad (8\text{-}142)$$
$$directZeroPredictionFlag = 1 \quad (8\text{-}143)$$

The process specified in subclause 8.4.1.2.1 is invoked with mbPartIdx, subMbPartIdx given as input and the output is assigned to refIdxCol and mvCol.

将 mbPartIdx, subMbPartIdx 作为输入，使用 8.4.1.2.1 小节中的处理过程，输出分配给 refIdxCol 和 mvCol。

The variable colZeroFlag is derived as follows.

变量 colZeroFlag 的计算如下：

- If all of the following conditions are true, colZeroFlag is set equal to 1.
- 如果下面的所有条件都成立，那么 colZeroFlag 将被设置成 1
  - the reference picture referred by RefPicList1[0] is a short-term reference picture
  - refIdxCol is equal to 0
  - both motion vector components mvCol[0] and mvCol[1] lie in the range of -1 to 1 (in quarter-sample units), inclusive
  - 参考图像 RefPicList1[0] 是一短期参考图像
  - refIdxCol 等于 0
  - 两个运动矢量分量mvCol[0] 和 mvCol[1] 都在[-1，1]内（四分之一象素精度）
- Otherwise, colZeroFlag is set equal to 0.
- 否则，colZeroFlag 被设置成等于 0

The motion vectors mvLX (with X being 0 or 1) are derived as follows.

运动矢量 mvLX （X 可能为 0 或者 1）的计算如下：

- If any of the following conditions is true, both components of the motion vector mvLX are set equal to 0.
- 如果下面的任何一个条件满足，那么运动矢量 mvLX 的两个分量都被设置成 0。
  - directZeroPredictionFlag is equal to 1
  - refIdxLX is less than 0
  - refIdxLX is equal to 0 and colZeroFlag is equal to 1

- directZeroPredictionFlag 等于 1
- refIdxLX 小于 0
- refIdxLX 等于 0 并且 colZeroFlag 等于 1

- Otherwise, the process specified in subclause 8.4.1.3 is invoked with mbPartIdx = 0, subMbPartIdx = 0, refIdxLX, and list suffix LX as the input and the output is assigned to mvLX.
- 否则，用 mbPartIdx = 0, subMbPartIdx = 0, refIdxLX, 和列表后缀 LX 作为输入，使用 8.4.1.3 小节中的处理过程，输出赋给 mvLX。

NOTE – In the immediately above case, the returned motion vector mvLX is identical for all 4x4 sub-macroblock partitions of a macroblock.
注 — 在上面的情况中，对于一个宏块的所有子宏块部分来说，返回的运动矢量 mvLX 都是相同的。

The prediction utilization flags predFlagL0 and predFlagL1 shall be derived as specified using Table 8-8.

预测使用标志predFlagL0 和 predFlagL1 的计算如 Table 8-8 所示。

Table 8-8 – Assignment of prediction utilization flags

refIdxL0	refIdxL1	predFlagL0	predFlagL1
>= 0	>= 0	1	1
>= 0	< 0	1	0
< 0	>= 0	0	1

8.4.1.2.3 Derivation process for temporal direct luma motion vector and reference index prediction mode

8.4.1.2.3 时间直接运动矢量和参考索引预测模式的处理过程

This process is invoked when direct_spatial_mv_pred_flag is equal to 0 and any of the following conditions is true.

当 direct_spatial_mv_pred_flag 等于 0 并且下面的任何一个条件正确时使用本过程。

- mb_type is equal to B_Skip
- mb_type is equal to B_Direct_16x16
- sub_mb_type[mbPartIdx] is equal to B_Direct_8x8.
- mb_type 等于 B_Skip
- B_Skip 等于B_Direct_16x16
- sub_mb_type[mbPartIdx] 等于 B_Direct_8x8

Inputs to this process are mbPartIdx and subMbPartIdx.

本过程的输入是 mbPartIdx 和 subMbPartIdx。

Outputs of this process are the motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, and the prediction list utilization flags, predFlagL0 and predFlagL1.
本过程的输出是运动矢量 mvL0 和 mvL1, 参考索引 refIdxL0 和 refIdxL1,和预测列表使用标志 predFlagL0 和 predFlagL1。
The process specified in subclause 8.4.1.2.1 is invoked with mbPartIdx, subMbPartIdx given as input and the output is assigned to colPic, mbAddrCol, mvCol, refIdxCol, and vertMvScale.
将 mbPartIdx, subMbPartIdx 作为输入，使用 8.4.1.2.1 小节的处理过程，输出分配给 colPic, mbAddrCol, mvCol, refIdxCol, 和 vertMvScale。

The reference indices refIdxL0 and refIdxL1 are derived as follows.

参考索引 refIdxL0 和 refIdxL1的计算如下：

refIdxL0 = ((refIdxCol < 0) ? 0 : MapColToList0(refIdxCol)) (8-144)

refIdxL1 = 0 (8-145)

NOTE - If the current macroblock is a field macroblock, refIdxL0 and refIdxL1 index a list of fields; otherwise (the current macroblock is a frame macroblock), refIdxL0 and refIdxL1 index a list of frames or complementary reference field pairs.

注 一 如果当前宏块是一场宏块，那么 refIdxL0 和 refIdxL1是一场列表的索引，否则（如果当前宏块是一帧宏块），那么 refIdxL0 和 refIdxL1 是帧列表和互补参考场对的索引。

The function MapColToList0(refIdxCol) is specified as follows.

函数 MapColToList0(refIdxCol) 的说明如下：

- Let refPicCol be a frame, a field, or a complementary field pair that was referred by the reference index refIdxCol when decoding the co-located macroblock mbAddrCol inside the picture colPic.
- 当解码图像 colPic 内对应位置宏块 mbAddrCol 时，用 refPicCol 表示由参考索引 refIdxCol 标识的一帧，一场，或者一互补场对。
- If vertMvScale is equal to One_To_One, MapColToList0(refIdxCol) returns the lowest valued reference index refIdxL0 in the current reference picture list RefPicList0 that references refPicCol. RefPicList0 shall contain a variable PicNum or LongTermPicNum that references refPicCol.
- 如果 vertMvScale 等于 One_To_One，那么 MapColToList0(refIdxCol) 返回包含 refPicCol 的当前参考图像列表 RefPicList0 中的最小的参考索引 refIdxL0。RefPicList0 应该包含 refPicCol 表示的变量PicNum 或者 LongTermPicNum。
- If vertMvScale is equal to Frm_To_Fld, MapColToList0(refIdxCol) returns the lowest valued reference index refIdxL0 in the current reference picture list RefPicList0 that references the field of refPicCol with the same parity as the current picture CurrPic. RefPicList0 shall contain a variable PicNum or LongTermPicNum that references the field of refPicCol with the same parity as the current picture CurrPic.
- 如果 vertMvScale 等于Frm_To_Fld，那么 MapColToList0(refIdxCol) 返回包含和当前图像 CurrPic 具有相同部分（底场或者顶场）的 refPicCol 的当前参考图像列表 RefPicList0中的最小参考索引 RefPicList0，RefPicList0 应该包含和当前图像具有相同部分（底场或者顶场）的 refPicCol 表示的变量 PicNum 或者 LongTermPicNum。
- Otherwise (vertMvScale is equal to Fld_To_Frm), MapColToList0(refIdxCol) returns the lowest valued reference index refIdxL0 in the current reference picture list RefPicList0 that references the frame or complementary field pair that contains refPicCol. RefPicList0 shall contain a variable PicNum or LongTermPicNum that references the frame or complementary field pair that contains refPicCol.
- 否则（vertMvScale 等于 Fld_To_Frm），MapColToList0(refIdxCol)返回包含帧和互补场对 refPicCol 的当前参考图像列表 RefPicList0中的最小参考索引 RefPicList0，RefPicList0 应该包含 refPicCol 表示的帧或参考场对的PicNum 或 LongTermPicNum。

  NOTE – A decoded reference picture that was marked as "used for short-term reference" when it was referenced in the decoding process of the picture containing the co-located macroblock may have been modified to be marked as "used for long-term reference" before being used for reference for inter prediction using the direct prediction mode for the current macroblock.

  注 一 在解码包含对应位置宏块的图像的过程中参考标识为“用作短期参考“的解码参考图像，在作为参考图像使用直接预测模式对当前宏块进行帧间预测之前，可以被修改为“用作长期参考”。（？？？？？）

Depending on the value of vertMvScale the vertical component of mvCol is modified as follows.

根据 vertMvScale 的值，mvCol 的垂直分量修改如下：

- If vertMvScale is equal to Frm_To_Fld

  如果 vertMvScale 等于 Frm_To_Fld

  mvCol[1] = mvCol[1] / 2 (8-146)

- If vertMvScale is equal to Fld_To_Frm

  如果 vertMvScale 等于 Fld_To_Frm

  mvCol[1] = mvCol[1] * 2 (8-147)

- Otherwise (vertMvScale is equal to One_To_One), mvCol[1] remains unchanged.

  否则（vertMvScale 等于 One_To_One），mvCol[1] 的值保持不变。

The two motion vectors mvL0 and mvL1 for each 4x4 sub-macroblock partition of the current macroblock are derived as follows:

当前宏块的每一个 4x4 子宏块部分的两个运动矢量 mvL0 和 mvL1的计算如下：

NOTE – It is often the case that many of the 4x4 sub-macroblock partitions share the same motion vectors and reference pictures. In these cases, temporal direct mode motion compensation can calculate the inter prediction sample values in larger units that 4x4 luma sample blocks. For example, when direct_8x8_inference_flag is equal to 1, at least each 8x8 luma sample quadrant of the macroblock shares the same motion vectors and reference pictures.

注 – 一般情况下，许多 4x4 子宏块部分使用相同的运动矢量和参考图像，因此时间直接模式运动补偿可以以大于 4x4 的亮度象素块来计算帧间预测值。例如：当 direct_8x8_inference_flag 等于 1 时，那么至少每个 8x8 的亮度象素块使用相同的运动矢量和参考图像。

– If the reference index refIdxL0 refers to a long-term picture, or DiffPicOrderCnt(picA, picB) with picA being the picture referred by RefPicList1[refIdxL1] and picB being the picture referred by RefPicList0[refIdxL0] is equal to 0, the motion vectors mvL0, mvL1 for the direct mode partition are derived by

– 如果参考索引 refIdxL0 指向一长期图像，或者 DiffPicOrderCnt(picA, picB) 等于 0，其中 RefPicList1[refIdxL1]表示的图像 picA 和 RefPicList0[refIdxL0] 表示的图像 picB 都等于 0，那么直接模式的运动矢量计算如下：

$$mvL0 = mvCol \quad (8\text{-}148)$$

$$mvL1 = 0 \quad (8\text{-}149)$$

– Otherwise, the motion vectors mvL0, mvL1 are derived as scaled versions of the motion vector mvCol of the co-located sub-macroblock partition as specified below (see Figure 8-2)

– 否则，使用对应位置子宏块部分的运动矢量 mvCol 来计算运动矢量 mvL0 和 mvL1，如下所示。

$$tx = (16\,384 + Abs(td / 2)) / td \quad (8\text{-}150)$$

$$DistScaleFactor = Clip3(-1024, 1023, (tb * tx + 32) >> 6) \quad (8\text{-}151)$$

$$mvL0 = (DistScaleFactor * mvCol + 128) >> 8 \quad (8\text{-}152)$$

$$mvL1 = mvL0 - mvCol \quad (8\text{-}153)$$

where tb and td are given as follows with pic0 being the decoded reference picture specified by RefPicList0[refIdxL0] and pic1 being the decoded reference picture specified by RefPicList1[refIdxL1]

这里，tb 和 td 的计算如下，pic0 是 RefPicList0[refIdxL0] 表示的解码参考图像，pic1 是 RefPicList1[refIdxL1] 表示的解码参考图像。

$$tb = Clip3(-128, 127, DiffPicOrderCnt(CurrPic, pic0)) \quad (8\text{-}154)$$

$$td = Clip3(-128, 127, DiffPicOrderCnt(pic1, pic0)) \quad (8\text{-}155)$$

NOTE - mvL0 and mvL1 cannot exceed the ranges specified in Annex A.

注 – mvL0 和 mvL1 不能够超出附录A中说明的范围。

The prediction utilization flags predFlagL0 and predFlagL1 are both set equal to 1.

预测使用标志 predFlagL0 和 predFlagL1 都被设置成 1。

Figure 8-2 illustrates the temporal direct-mode motion vector inference when the current picture is temporally between the list 0 reference picture and the list 1 reference picture.

Figure 8-2 表示了时间直接模式运动矢量的计算，这里当前图像在列表 0 参考图像和列表 1 参考图像之间。

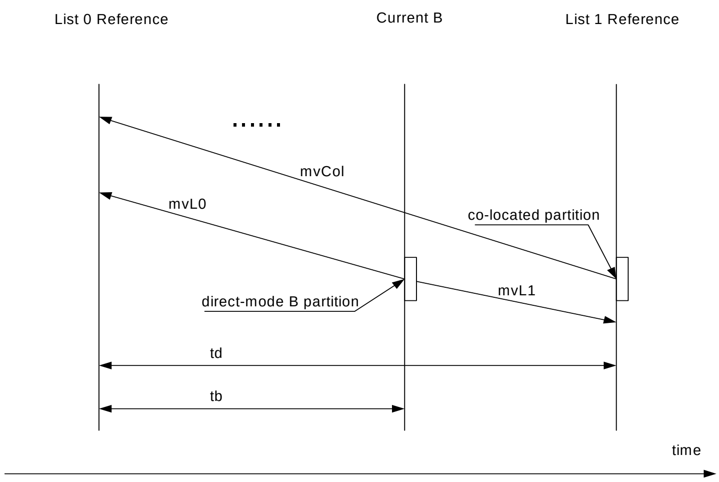


Figure 8-2 –Example for temporal direct-mode motion vector inference (informative)

8.4.1.3 Derivation process for luma motion vector prediction

8.4.1.3 亮度运动矢量预测的过程

Inputs to this process are

本过程的输入是：

- the macroblock partition index mbPartIdx,
- the sub-macroblock partition index subMbPartIdx,
- list suffix LX,
- the reference index of the current partition refIdxLX.

- 宏块部分索引 mbPartIdx
- 子宏块部分索引 subMbPartIdx
- 列表后缀 LX
- 当前部分的参考索引 refIdxLX

Output of this process is the prediction mvpLX of the motion vector mvLX.

本过程的输出是运动矢量 mvLX 的预测值 mvpLX。

The derivation process for the neighbouring blocks for motion data in subclause 8.4.1.3.2 is invoked with mbPartIdx, subMbPartIdx, and list suffix LX as the input and with mbAddrN\mbPartIdxN\subMbPartIdxN, reference indices refIdxLXN and the motion vectors mvLXN with N being replaced by A, B, or C as the output.

将 mbPartIdx, subMbPartIdx, 和列表后缀 LX 作为输入，使用 8.4.1.3.2 小节中的关于运动矢量数据相邻块的处理过程，然后输出 mbAddrN\mbPartIdxN\subMbPartIdxN，参考索引 refIdxLXN 和运动矢量 mvLXN（N 可能为A、B 或者 C）。

The derivation process for median luma motion vector prediction in subclause 8.4.1.3.1 is invoked with mbAddrN\mbPartIdxN\subMbPartIdxN, mvLXN, refIdxLXN with N being replaced by A, B, or C and refIdxLX as the input and mvpLX as the output, unless one of the following is true.

将 mbAddrN\mbPartIdxN\subMbPartIdxN, mvLXN, refIdxLXN（N 可能为A、B 或者 C）和 refIdxLX 作为输入，使用 8.4.1.3.1 小节中的中值亮度运动矢量预测的处理过程，计算出 mvpLX 并且输出，除非下面的一个条件成立。

- MbPartWidth(mb_type) is equal to 16, MbPartHeight(mb_type) is equal to 8, mbPartIdx is equal to 0, and refIdxLXB is equal to refIdxLX,

- MbPartWidth(mb_type) 等于 16，MbPartHeight(mb_type) 等于 8，mbPartIdx 等于 0，和 refIdxLXB 等于 refIdxLX，

mvpLX = mvLXB (8-156)

- MbPartWidth(mb_type) is equal to 16, MbPartHeight(mb_type) is equal to 8, mbPartIdx is equal to 1, and refIdxLXA is equal to refIdxLX,
- MbPartWidth(mb_type) 等于 16，MbPartHeight(mb_type) 等于 8，mbPartIdx 等于 1，和 refIdxLXA 等于 refIdxLX，

mvpLX = mvLXA (8-157)

- MbPartWidth(mb_type) 等于 8，MbPartHeight(mb_type) 等于 16，mbPartIdx 等于 0，和 refIdxLXA 等于 refIdxLX，

mvpLX = mvLXA (8-158)

- MbPartWidth(mb_type) 等于 8，MbPartHeight(mb_type) 等于 16，mbPartIdx 等于 1，和 refIdxLXC 等于 refIdxLX，

mvpLX = mvLXC (8-159)

Figure 8-3 illustrates the non-median prediction as described above.

Figure 8-3 说明了上述描述的非中值预测

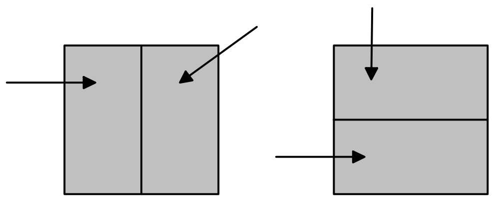


Figure 8-3 – Directional segmentation prediction (informative)

8.4.1.3.1 Derivation process for median luma motion vector prediction

8.4.1.3.1 中值亮度运动矢量预测的处理过程

Inputs to this process are

本过程的输入是：

- the neighbouring partitions mbAddrN\mbPartIdxN\subMbPartIdxN (with N being replaced by A, B, or C),
- the motion vectors mvLXN (with N being replaced by A, B, or C) of the neighbouring partitions,
- the reference indices refIdxLXN (with N being replaced by A, B, or C) of the neighbouring partitions, and
- the reference index refIdxLX of the current partition.
- 相邻部分 mbAddrN\mbPartIdxN\subMbPartIdxN （N 可能为A、B 或者 C），
- 相邻部分的运动矢量mvLXN（N 可能为A、B 或者 C），
- 相邻部分的参考索引refIdxLXN（N 可能为A、B 或者 C），
- 当前部分的参考索引 refIdxLX。

Output of this process is the motion vector prediction mvpLX.

本过程的输出是运动矢量预测 mvpLX。

The variable mvpLX is derived as follows:

变量 mvpLX 的计算如下：

- When both partitions mbAddrB\mbPartIdxB\subMbPartIdxB and mbAddrC\mbPartIdxC\subMbPartIdxC are not available and mbAddrA\mbPartIdxA\subMbPartIdxA is available,
- 当 mbAddrB\mbPartIdxB\subMbPartIdxB 和 mbAddrC\mbPartIdxC\subMbPartIdxC 都不可用，但是 mbAddrA\mbPartIdxA\subMbPartIdxA 可用时，那么

mvLXB = mvLXA (8-160)

mvLXC = mvLXA (8-161)

refIdxLXB = refIdxLXA (8-162)

refIdxLXC = refIdxLXA (8-163)

- Depending on reference indices refIdxLXA, refIdxLXB, or refIdxLXC, the following applies
- 根据参考索引 refIdxLXA, refIdxLXB, 或者 refIdxLXC 的值，使用下面的过程
  - If one and only one of the reference indices refIdxLXA, refIdxLXB, or refIdxLXC is equal to the reference index refIdxLX of the current partition, the following applies. Let refIdxLXN be the reference index that is equal to refIdxLX, the motion vector mvLXN is assigned to the motion vector prediction mvpLX:
  - 如果在参考索引refIdxLXA, refIdxLXB, 或者 refIdxLXC中有且仅有一个参考索引等于当前部分的参考索引refIdxLX，那么使用下面的过程。将 refIdxLX设置成等于 refIdxLXN 的参考索引，运动矢量 mvLXN 赋给运动矢量预测 mvpLX。

  mvpLX = mvLXN (8-164)

  - Otherwise, each component of the motion vector prediction mvpLX is given by the median of the corresponding vector components of the motion vector mvLXA, mvLXB, and mvLXC:
  - 否则，运动矢量预测值 mvpLX 的每一个分量等于运动矢量 mvLXA, mvLXB, 和 mvLXC相对应分量的中值。

  mvpLX[0] = Median(mvLXA[0], mvLXB[0], mvLXC[0]) (8-165)

  mvpLX[1] = Median(mvLXA[1], mvLXB[1], mvLXC[1]) (8-166)

8.4.1.3.2 Derivation process for motion data of neighbouring partitions

8.4.1.3.2 相邻部分运动数据的处理过程

Inputs to this process are

本过程的输入是：

- the macroblock partition index mbPartIdx,
- the sub-macroblock partition index subMbPartIdx,
- the list suffix LX
- 宏块部分索引 mbPartIdx，
- 子宏块部分索引 subMbPartIdx，
- 列表后缀 LX

Outputs of this process are (with N being replaced by A, B, or C)

本过程的输出是（N 可能为A、B 或者 C）

- mbAddrN\mbPartIdxN\subMbPartIdxN specifying neighbouring partitions,
- the motion vectors mvLXN of the neighbouring partitions, and

- the reference indices refIdxLXN of the neighbouring partitions.
- 相邻的部分 mbAddrN\mbPartIdxN\subMbPartIdxN，
- 相邻部分的运动矢量 mvLXN，
- 相邻部分的参考索引 refIdxLXN。

The partitions mbAddrN\mbPartIdxN\subMbPartIdxN with N being either A, B, or C are derived in the following ordered steps.

相邻部分 mbAddrN\mbPartIdxN\subMbPartIdxN（N 可能为A、B 或者 C）的计算如下：

1. Let mbAddrD\mbPartIdxD\subMbPartIdxD be variables specifying an additional neighbouring partition.

   假设 mbAddrD\mbPartIdxD\subMbPartIdxD 为另外一个相邻部分的变量。

2. The process in subclause 6.4.7.5 is invoked with mbPartIdx and subMbPartIdx as input and the output is assigned to mbAddrN\mbPartIdxN\subMbPartIdxN with N being replaced by A, B, C, or D.

   将 mbPartIdx 和 subMbPartIdx 作为输入，使用 6.4.7.5 小节中的处理过程，输出 mbAddrN\mbPartIdxN\subMbPartIdxN（N 可能为A、B 、 C或 D）。

3. When the partition mbAddrC\mbPartIdxC\subMbPartIdxC is not available, the following applies

   当 mbAddrC\mbPartIdxC\subMbPartIdxC 不可用时，使用下面的过程，

$$mbAddrC = mbAddrD \tag{8-167}$$

$$mbPartIdxC = mbPartIdxD \tag{8-168}$$

$$subMbPartIdxC = subMbPartIdxD \tag{8-169}$$

The motion vectors mvLXN and reference indices refIdxLXN (with N being A, B, or C) are derived as follows.

运动矢量 mvLXN 和参考索引 refIdxLXN （N 可能为A、B 或者 C）的计算如下：

- If the macroblock partition or sub-macroblock partition mbAddrN\mbPartIdxN\subMbPartIdxN is not available or mbAddrN is coded in Intra prediction mode or predFlagLX of mbAddrN\mbPartIdxN\subMbPartIdxN is equal to 0, both components of mvLXN are set equal to 0 and refIdxLXN is set equal to –1.
- 如果宏块部分或者子宏块部分 mbAddrN\mbPartIdxN\subMbPartIdxN 不可用，或者 mbAddrN 是帧内预测编码模式，或者 mbAddrN\mbPartIdxN\subMbPartIdxN 的 predFlagLX 等于 0，那么 mvLXN 的两个分量都设置成 0 并且 refIdxLXN 设置成 –1。
- Otherwise, the following applies.
- 否则，使用下面的过程
  - The motion vector mvLXN and reference index refIdxLXN are set equal to MvLX[mbPartIdxN][subMbPartIdxN] and RefIdxLX[mbPartIdxN], respectively, which are the motion vector mvLX and reference index refIdxLX that have been assigned to the (sub-)macroblock partition mbAddrN\mbPartIdxN\subMbPartIdxN.
  - 运动矢量 mvLXN 和 参考索引 refIdxLXN 都被分别设置成等于 MvLX[mbPartIdxN][subMbPartIdxN] 和 RefIdxLX[mbPartIdxN]，它们是（子）宏块部分 mbAddrN\mbPartIdxN\subMbPartIdxN 的运动矢量 mvLX 和参考索引 refIdxLX。
  - If the current macroblock is a field macroblock and the macroblock mbAddrN is a frame macroblock
  - 如果当前宏块是场宏块并且宏块mbAddrN是帧宏块，那么

$$mvLXN[\ 1\ ] = mvLXN[\ 1\ ] / 2 \tag{8-170}$$

$$refIdxLXN = refIdxLXN * 2 \tag{8-171}$$

  - If the current macroblock is a frame macroblock and the macroblock mbAddrN is a field macroblock
  - 如果当前宏块是帧宏块并且宏块 mbAddrN 是场宏块，，那么

mvLXN[1] = mvLXN[1] * 2 (8-172)

refIdxLXN = refIdxLXN / 2 (8-173)

- Otherwise, the vertical motion vector component mvLXN[1] and the reference index refIdxLXN remain unchanged.
- 否则，垂直运动矢量成分 mvLXN[1] 和 参考索引 refIdxLXN 保持不变。

8.4.1.4 Derivation process for chroma motion vectors

8.4.1.4 色差运动矢量的处理过程

Inputs to this process are a luma motion vector mvLX and a reference index refIdxLX.

本过程的输入是亮度的运动矢量 mvLX 和参考索引 refIdxLX。

Outputs of this process are a chroma motion vector mvCLX.

本过程的输出是色差运动矢量 mvCLX。

A chroma motion vector is derived from the corresponding luma motion vector. Since the accuracy of luma motion vectors is one-quarter sample and chroma has half resolution compared to luma, the accuracy of chroma motion vectors is one-eighth sample, i.e., a value of 1 for the chroma motion vector refers to a one-eighth sample displacement.

色差的运动矢量是由对应亮度的运动矢量计算得到的。因为亮度运动矢量的精度是1/4象素而且色差的分辨率只有亮度分辨率的一半，因此色差运动矢量的精度是1/8象素，例如，一个单位的色差运动矢量表示1/8的象素位移。

NOTE - For example when the luma vector applies to 8x16 luma samples, the corresponding chroma vector applies to 4x8 chroma samples and when the luma vector applies to 4x4 luma samples, the corresponding chroma vector applies to 2x2 chroma samples.

注 － 例如当亮度运动矢量应用于 8x16 的亮度象素时，那么对应的色差运动矢量就应用于 4x8 的色差象素，当亮度运动矢量应用于 4x4 的亮度象素时，那么对应的色差运动矢量就应用于 2x2 的色差象素

For the derivation of the motion vector mvCLX, the following applies.

对于运动矢量 mvCLX 的计算，使用下面的过程。

- If the current macroblock is a frame macroblock, the horizontal and vertical components of the chroma motion vector mvCLX are derived by multiplying the corresponding components of luma motion vector mvLX by 2, through mapping one-quarter sample mvLX units to one-eighth sample mvCLX units
- 如果当前宏块是一帧宏块，那么色差运动矢量 mvCLX 的垂直分量和水平分量由亮度运动矢量 mvLX 的对应分量乘以 2 得到，从1/4象素精度的 mvLX 映射到 1/8 象素精度的mvCLX。

mvCLX[0] = mvLX[0] (8-174)
mvCLX[1] = mvLX[1] (8-175)

- Otherwise (the current macroblock is a field macroblock), only the horizontal component of the chroma motion vector mvCLX[0] is derived using Equation 8-174. The vertical component of the chroma motion vector mvCLX[1] is dependent on the parity of the current field or the current macroblock and the reference picture, which is referred by the reference index refIdxLX. mvCLX[1] is derived from mvLX[1] according to Table 8-9.
- 否则（当前宏块为场宏块），只有色差运动矢量的水平分量 mvCLX[0] 使用等式 8-174 计算得到。色差运动矢量的垂直分量 mvCLX[1] 则根据当前场或当前宏块和参考图像的场次（顶场还是底场）计算得到，参考图像由参考索引 refIdxLX 标识，mvCLX[1] 根据 Table 8-9 由 mvLX[1] 计算得到。（具体的原因需要介绍清楚）

Table 8-9 – Derivation of the vertical component of the chroma vector in field coding mode

Parity conditions		**mvCLX[1]**
Reference picture (refIdxLX)	Current field (picture/macroblock)	
Top field	Bottom field	mvLX[1] + 2
Bottom field	Top field	mvLX[1] – 2

Otherwise	mvLX[1]

8.4.2 Decoding process for Inter prediction samples

8.4.2 帧间预测象素的解码过程

Inputs to this process are

本过程的输入是:

- a macroblock partition mbPartIdx,
- a sub-macroblock partition subMbPartIdx.
- variables specifying partition width and height, partWidth and partHeight
- luma motion vectors mvL0 and mvL1 and chroma motion vectors mvCL0 and mvCL1
- reference indices refIdxL0 and refIdxL1
- prediction list utilization flags, predFlagL0 and predFlagL1
- 一宏块部分索引 mbPartIdx,
- 一子宏块部分索引 subMbPartIdx,
- 表示(子)宏块部分的宽度和高度变量 partWidth 和 partHeight,
- 亮度运动矢量 mvL0、mvL1 和色差运动矢量 mvCL0 、mvCL1,
- 参考索引 refIdxL0 和 refIdxL1,
- 参考列表使用标志 predFlagL0 和 predFlagL1

Outputs of this process are

本过程的输出是:

- the Inter prediction samples predPart, which are a (partWidth)x(partHeight) array $predPart_L$ of prediction luma samples, and two (partWidth/2)x(partHeight/2) arrays $predPart_{Cb}$, $predPart_{Cr}$ of prediction chroma samples, one for each of the chroma components Cb and Cr.
- 帧间预测象素 predPart,包括一(partWidth)x(partHeight)预测亮度象素数组 $predPart_L$ 和两个(partWidth/2)x(partHeight/2)预测色差象素数组,分别对应于色差分量 Cb 和 Cr。

Let $predPartL0_L$ and $predPartL1_L$ be (partWidth)x(partHeight) arrays of predicted luma sample values and $predPartL0_{Cb}$, $predPartL1_{Cb}$, $predPartL0_{Cr}$, and $predPartL1_{Cr}$ be (partWidth/2)x(partHeight/2) arrays of predicted chroma sample values.

假设 $predPartL0_L$ 和 $predPartL1_L$ 为(partWidth)x(partHeight)预测亮度象素值数组,$predPartL0_{Cb}$, $predPartL1_{Cb}$, $predPartL0_{Cr}$, 和 $predPartL1_{Cr}$为 (partWidth/2)x(partHeight/2) 预测色差象素值数组。

For LX being replaced by either L0 or L1 in the variables predFlagLX, RefPicListX, refIdxLX, refPicLX, predPartLX, the following is specified.

在变量 predFlagLX, RefPicListX, refIdxLX, refPicLX, predPartLX 中,LX 可能是 L0 或者 L1。

When predFlagLX is equal to 1, the following applies.

当 predFlagLX 等于 1 时,使用下面的过程。

- The reference frame consisting of an ordered two-dimensional array $refPicLX_L$ of luma samples and two ordered two-dimensional arrays $refPicLX_{Cb}$ and $refPicLX_{Cr}$ of chroma samples is derived by invoking the process specified in subclause 8.4.2.1 with refIdxLX and RefPicListX given as input.
- The arrays $predPartLX_L$, $predPartLX_{Cb}$, and $predPartLX_{Cr}$ are derived by invoking the process specified in subclause 8.4.2.2 with the current partition specified by mbPartIdx\subMbPartIdx, the motion vectors mvLX, mvCLX, and the reference arrays with $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$ given as input.
- 将 refIdxLX 和 RefPicListX 作为输入,使用 8.4.2.1 小节中的处理过程,计算出参考图像并输出,该参考图像由一个二维亮度象素数组$refPicLX_L$和两个二维色差象素数组 $refPicLX_{Cb}$ 和 $refPicLX_{Cr}$ 组成。

- 将mbPartIdx\subMbPartIdx，运动矢量mvLX, mvCLX,和参考图像 refPicLX$_L$, refPicLX$_{Cb}$, and refPicLX$_{Cr}$作为输入，使用 8.4.2.2 小节中的处理过程，计算出predPartLX$_L$, predPartLX$_{Cb}$, 和 predPartLX$_{Cr}$并输出。

For C being replaced by L, Cb, or Cr, the array predPart$_C$ of the prediction samples of component C is derived by invoking the process specified in subclause 8.4.2.3 with the current partition specified by mbPartIdx and subMbPartIdx and the array predPartL0$_C$ and predPartL1$_C$ as well as predFlagL0 and predFlagL1 given as input.

将由mbPartIdx 和 subMbPartIdx表示的当前部分，数组predPartL0$_C$ 和 predPartL1$_C$ 和predFlagL0 and predFlagL1作为输入，其中C 可能是 L, Cb, 或者 Cr，使用 8.4.2.3 小节中的处理过程，计算出当前预测成分 C的预测象素值predPart$_C$。

8.4.2.1 Reference picture selection process

8.4.2.1 参考图像的选择过程

Input to this process is a reference index refIdxLX.

本过程的输入是参考索引 refIdxLX。

Output of this process is a reference picture consisting of a two-dimensional array of luma samples refPicLX$_L$ and two two-dimensional arrays of chroma samples refPicLX$_{Cb}$ and refPicLX$_{Cr}$.

本过程的输出是一参考图像，包括一二维亮度象素数组 refPicLX$_L$和二个二维色差象素数组refPicLX$_{Cb}$ 和 refPicLX$_{Cr}$。

Reference picture list RefPicListX is a list of variables PicNum (for short-term reference pictures) and LongTermPicNum (for long-term reference pictures) of previously decoded reference frames, complementary reference field pairs, or non-paired reference fields that have been marked as "used for reference" as specified in subclause 8.2.5.

参考图像列表 RefPicListX 是变量 PicNum（对于短期参考图像）和 LongTermPicNum（对于长期参考图像）的列表，这些参考图像是 8.2.5 小节中说明"用作参考"的解码参考帧、互补参考场对和非成对的参考场。

- If field_pic_flag is equal to 1, all entries of the RefPicListX are variables PicNum and LongTermPicNum of decoded reference fields or fields of decoded reference frames.
- Otherwise (field_pic_flag is equal to 0), all entries of RefPicListX are variables PicNum and LongTermPicNum of decoded reference frames or complementary reference field pairs.
- 如果 field_pic_flag 等于 1，那么 RefPicListX 中的所有索引都是解码参考场或解码参考帧的场的PicNum 和 LongTermPicNum。
- 否则（field_pic_flag 等于 0），那么RefPicListX 中的所有索引都是解码参考帧或互补参考场对的PicNum 和 LongTermPicNum。

The reference picture list RefPicListX is derived as specified in subclause 8.2.4.

参考图像列表 RefPicListX 的 计算如 8.2.4 小节所示。

For the derivation of the reference picture, the following applies.

对于参考图像的计算，使用下面的过程。

- If field_pic_flag is equal to 1, the reference field or field of a reference frame referred by PicNum = RefPicListX[refIdxLX] or LongTermPicNum = RefPicListX[refIdxLX] shall be the output. The output reference field or field of a reference frame consists of a (PicWidthInSamples$_L$)x(PicHeightInSamples$_L$) array of luma samples refPicLX$_L$ (equal to SL of) and two (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$) arrays of chroma samples refPicLX$_{Cb}$ and refPicLX$_{Cr}$.
- 如果 field_pic_flag 等于 1，那么输出由 PicNum = RefPicListX[refIdxLX] 或 LongTermPicNum = RefPicListX[refIdxLX]表示的参考场或一参考帧的场，该参考场或参考帧的场由一(PicWidthInSamples$_L$)x(PicHeightInSamples$_L$) 亮度象素数组 refPicLX$_L$ 和两个(PicWidthInSamples$_C$)x(PicHeightInSamples$_C$) 色差象素数组refPicLX$_{Cb}$ 和 refPicLX$_{Cr}$组成。
- Otherwise (field_pic_flag is equal to 0), the following applies.
- 否则（field_pic_flag 等于 0），使用下面的过程
  - If the current macroblock is a frame macroblock, the reference frame or complementary field pair referred by PicNum = RefPicListX[refIdxLX] or LongTermPicNum = RefPicListX[refIdxLX] shall be the output. The output reference frame or complementary reference field pair consists of a

(PicWidthInSamples$_L$)x(PicHeightInSamples$_L$) array of luma samples refPicLX$_L$ and two (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$) arrays of chroma samples refPicLX$_{Cb}$ and refPicLX$_{Cr}$.

- 如果当前宏块是帧宏块，那么应该输出 由 PicNum = RefPicListX[refIdxLX] 或 LongTermPicNum = RefPicListX[refIdxLX] 表示的参考帧或者互补场对。这个参考帧或互补的参考场对由一个 (PicWidthInSamples$_L$)x(PicHeightInSamples$_L$) 亮度象素数组 refPicLX$_L$ 和两个 (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$) 色差象素数组refPicLX$_{Cb}$ 和 refPicLX$_{Cr}$ 组成。

- Otherwise (the current macroblock is a field macroblock), the following applies.

- 否则（当前宏块是一场宏块），那么使用下面的过程。

  - Let refFrame be the reference frame or complementary reference field pair that is referred by PicNum = RefPicListX[refIdxLX / 2] or LongTermPicNum = RefPicListX[refIdxLX / 2].

  - 假设 refFrame 是由 PicNum = RefPicListX[refIdxLX / 2] 或者 LongTermPicNum = RefPicListX[refIdxLX / 2] 表示的参考帧或互补的参考场对。

  - The field of refFrame is selected as follows.

  - 选择 refFrame 的场如下：

    - If refIdxLX % 2 is equal to 0, the field of refFrame that has the same parity as the current macroblock shall be the output.

    - Otherwise (refIdxLX % 2 is equal to 1), the field of refFrame that has the opposite parity as the current macroblock shall be the output.

    - 如果 refIdxLX % 2 等于 0，那么就输出和当前宏块具有相同场的refFrame的场。

    - 否则（refIdxLX % 2 is equal to 1），那么就输出和当前宏块具有相对场的refFrame的场。

  - The output reference field or field of a reference frame consists of a (PicWidthInSamples$_L$)x(PicHeightInSamples$_L$/2) array of luma samples refPicLX$_L$ and two (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$/2) arrays of chroma samples refPicLX$_{Cb}$ and refPicLX$_{Cr}$.

  - 输出的参考场或者一参考帧的场由一(PicWidthInSamples$_L$)x(PicHeightInSamples$_L$/2)亮度象素数组 refPicLX$_L$ 和两个 (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$/2) 色差象素数组refPicLX$_{Cb}$ 和 refPicLX$_{Cr}$ 组成。

8.4.2.2 Fractional sample interpolation process

8.4.2.2 分象素内插处理过程

Inputs to this process are

本过程的输入是：

- the current partition given by its partition index mbPartIdx and its sub-macroblock partition index subMbPartIdx,
- the width and height partWidth, partHeight of this partition in luma-sample units,
- a luma motion vector mvLX given in quarter-luma-sample units,
- a chroma motion vector mvCLX given in eighth-chroma-sample units, and
- the selected reference picture sample arrays refPicLX$_L$, refPicLX$_{Cb}$, and refPicLX$_{Cb}$
- 由宏块部分索引 mbPartIdx 和子宏块部分索引 subMbPartIdx 表示的当前部分，
- 当前部分亮度象素的宽 partWidth 与高 partHeight，
- 亮度分量1/4象素精度运动矢量 mvLX，
- 色差分量1/8象素精度运动矢量 mvCLX，
- 选择的参考图像 refPicLX$_L$, refPicLX$_{Cb}$, 和 refPicLX$_{Cb}$，

Outputs of this process are

本过程的输出是：

- a (partWidth)x(partHeight) array predPartLX$_L$ of prediction luma sample values and

- two (partWidth/2)x(partHeight/2) arrays $predPartLX_{Cb}$, and $predPartLX_{Cr}$ of prediction chroma sample values.
- 一个 (partWidth)x(partHeight) 预测亮度象素数组 $predPartLX_L$，
- 两个(partWidth/2)x(partHeight/2)预测色差象素数组 $predPartLX_{Cb}$ 和 $predPartLX_{Cr}$

Let (xA_L, yA_L) be the location given in full-sample units of the upper-left luma sample of the current partition given by mbPartIdx\subMbPartIdx relative to the upper-left luma sample location of the given two-dimensional array of luma samples.

假设 (xA_L, yA_L) 为当前 mbPartIdx\subMbPartIdx 部分的左上角亮度象素相对于给定二维的亮度分量左上角象素的整象素坐标位置。

Let ($xInt_L$, $yInt_L$) be a luma location given in full-sample units and ($xFrac_L$, $yFrac_L$) be an offset given in quarter-sample units. These variables are used only inside this subclause for specifying general fractional-sample locations inside the reference sample arrays $refPicLX_L$, $refPicLX_{Cb}$, and $refPicLX_{Cr}$.

假设 ($xInt_L$, $yInt_L$) 为整象素亮度位置坐标，($xFrac_L$, $yFrac_L$)为1/4象素精度的偏移坐标。在参考象素数组 $refPicLX_L$, $refPicLX_{Cb}$, 和 $refPicLX_{Cr}$中，为了表示一般的分象素坐标才使用这些变量。

For each luma sample location ($0 <= x_L < partWidth$, $0 <= y_L < partHeight$) inside the prediction luma sample array $predLX_L$, the corresponding predicted luma sample value $predLX_L[x_L, y_L]$ is derived as follows:

在预测亮度数组 $predLX_L$ 中对于每一个亮度象素位置 ($0 <= x_L < partWidth$, $0 <= y_L < partHeight$)，对应的预测亮度象素值 $predLX_L[x_L, y_L]$ 的计算如下：

$$xInt_L = xA_L + (mvLX[0] >> 2) + x_L \quad (8\text{-}176)$$
$$yInt_L = yA_L + (mvLX[1] >> 2) + y_L \quad (8\text{-}177)$$

$$xFrac_L = mvLX[0] \& 3 \quad (8\text{-}178)$$
$$yFrac_L = mvLX[1] \& 3 \quad (8\text{-}179)$$

- The prediction sample value $predLX_L[x_L, y_L]$ is derived by invoking the process specified in subclause 8.4.2.2.1 with ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) and $refPicLX_L$ given as input.
- 将 ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) 和 $refPicLX_L$作为输入，使用 8.4.2.2.1 小节中的处理过程，计算出预测象素值 $predLX_L[x_L, y_L]$。

Let ($xInt_C$, $yInt_C$) be a chroma location given in full-sample units and ($xFrac_C$, $yFrac_C$) be an offset given in one-eighth sample units. These variables are used only inside this subclause for specifying general fractional-sample locations inside the reference sample arrays $refPicLX_{Cb}$, and $refPicLX_{Cr}$.

假设 ($xInt_C$, $yInt_C$) 是整象素色差位置，($xFrac_C$, $yFrac_C$) 为1/8象素精度的偏移坐标，在参考象素数组$refPicLX_{Cb}$, 和 $refPicLX_{Cr}$中为了表示一般的分象素坐标位置才使用这些变量。

For each chroma sample location ($0 <= x_C < partWidth/2$, $0 <= y_C < partHeight/2$) inside the prediction chroma sample arrays $predPartLX_{Cb}$ and $predPartLX_{Cr}$, the corresponding prediction chroma sample values $predPartLX_{Cb}[x_C, y_C]$ and $predPartLX_{Cr}[x_C, y_C]$ are derived as follows:

在预测色差象素数组 $predPartLX_{Cb}$ 和 $predPartLX_{Cr}$ 中对于每一个色差象素位置 ($0 <= x_C < partWidth/2$, $0 <= y_C < partHeight/2$)，对应的预测色差象素值 $predPartLX_{Cb}[x_C, y_C]$ 和 $predPartLX_{Cr}[x_C, y_C]$的计算如下：

$$xInt_C = (xA_L >> 1) + (mvCLX[0] >> 3) + x_C \quad (8\text{-}180)$$
$$yInt_C = (yA_L >> 1) + (mvCLX[1] >> 3) + y_C \quad (8\text{-}181)$$

$$xFrac_C = mvCLX[0] \& 7 \quad (8\text{-}182)$$
$$yFrac_C = mvCLX[1] \& 7 \quad (8\text{-}183)$$

- The prediction sample value $predPartLX_{Cb}[x_C, y_C]$ is derived by invoking the process specified in subclause 8.4.2.2.2 with ($xInt_C$, $yInt_C$), ($xFrac_C$, $yFrac_C$) and $refPicLX_{Cb}$ given as input.
- The prediction sample value $predPartLX_{Cr}[x_C, y_C]$ is derived by invoking the process specified in subclause 8.4.2.2.2 with ($xInt_C$, $yInt_C$), ($xFrac_C$, $yFrac_C$) and $refPicLX_{Cr}$ given as input.
- 将 ($xInt_C$, $yInt_C$), ($xFrac_C$, $yFrac_C$) 和 $refPicLX_{Cb}$ 作为输入，使用 8.4.2.2.2 小节中的处理过程，计算出预测象素值 $predPartLX_{Cb}[x_C, y_C]$。

- 将 ($xInt_C$, $yInt_C$), ($xFrac_C$, $yFrac_C$) 和 $refPicLX_{Cr}$ 作为输入，使用 8.4.2.2.2 小节中的处理过程，计算出预测象素值 $predPartLX_{Cr}[x_C, y_C]$。

8.4.2.2.1 Luma sample interpolation process

8.4.2.2.1 亮度象素内插处理过程

Inputs to this process are

本过程的输入是：

- a luma location in full-sample units ($xInt_L$, $yInt_L$),
- a luma location offset in fractional-sample units ($xFrac_L$, $yFrac_L$), and
- the luma sample array of the selected reference picture $refPicLX_L$
- 整象素亮度位置坐标 ($xInt_L$, $yInt_L$)，
- 分象素亮度位置偏移坐标 ($xFrac_L$, $yFrac_L$)，
- 选择的参考图像亮度象素数组 $refPicLX_L$

Output of this process is a predicted luma sample value $predPartLX_L[x_L, y_L]$.

本过程的输出是预测亮度象素值$predPartLX_L[x_L, y_L]$。

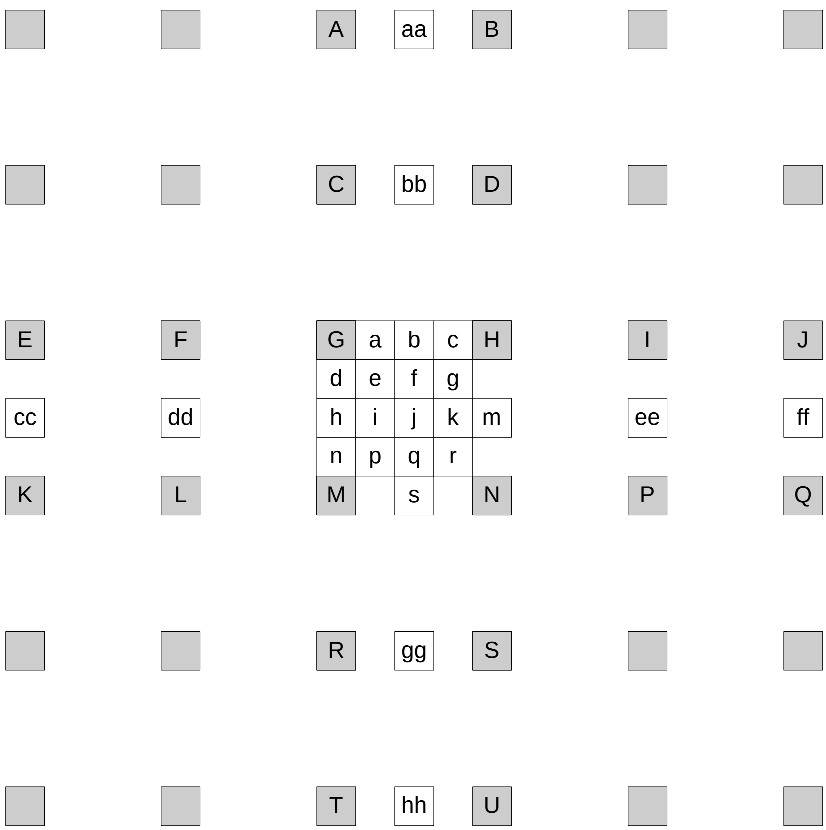


Figure 8-4 – Integer samples (shaded blocks with upper-case letters) and fractional sample positions (un-shaded blocks with lower-case letters) for quarter sample luma interpolation.

In Figure 8-4, the positions labelled with upper-case letters within shaded blocks represent luma samples at full-sample locations inside the given two-dimensional array $refPicLX_L$ of luma samples. These samples may be used for generating the predicted luma sample value $predPartLX_L[x_L, y_L]$. The locations (xZ_L, yZ_L) for each of the corresponding luma samples Z, where Z may be A, B, C, D, E, F, G, H, I, J, K, L, M, N, P, Q, R, S, T, or U, inside the given array $refPicLX_L$ of luma samples are derived as follows:

在图 Figure 8-4 中，阴影块中标有大写字母的位置表示给定的二维亮度象素数组 $refPicLX_L$中亮度整象素，这些象素也许被用来产生预测的亮度象素值 $predPartLX_L[x_L, y_L]$。在给定的亮度数组 $refPicLX_L$ 中，对于每一个对应亮度象素 Z，Z 可能为A, B, C, D, E, F, G, H, I, J, K, L, M, N, P, Q, R, S, T, 或 U,，位置(xZ_L, yZ_L)的计算如下：

$$xZ_L = \text{Clip3}(0, \text{PicWidthInSamples}_L - 1, \text{xInt}_L + xDZ_L)$$
$$yZ_L = \text{Clip3}(0, \text{PicHeightInSamples}_L - 1, \text{yInt}_L + yDZ_L) \quad (8\text{-}184)$$

Table 8-10 specifies (xDZ_L, yDZ_L) for different replacements of Z.

对于 Z 的不同偏移，Table 8-10 说明了 (xDZ_L, yDZ_L)

Table 8-10 – Differential full-sample luma locations

Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	U
xDZ_L	0	1	0	1	-2	-1	0	1	2	3	-2	-1	0	1	2	3	0	1	0	1
yDZ_L	-2	-2	-1	-1	0	0	0	0	0	0	1	1	1	1	1	1	2	2	3	3

Given the luma samples 'A' to 'U' at full-sample locations (xA_L, yA_L) to (xU_L, yU_L), the luma samples 'a' to 's' at fractional sample positions are derived by the following rules. The luma prediction values at half sample positions shall be derived by applying a 6-tap filter with tap values (1, -5, 20, 20, -5, 1). The luma prediction values at quarter sample positions shall be derived by averaging samples at full and half sample positions. The process for each fractional position is described below.

根据位置为(xA_L, yA_L) 到(xU_L, yU_L)的亮度整象素 'A' 到 'U'，分象素位置的亮度象素'a' 到's'可以通过下面的规则计算出来。半象素位置的亮度预测值可以通过使用6 阶滤波器(1, -5, 20, 20, -5, 1)计算得到，1/4象素位置的亮度预测值可以通过平均整象素位置和半象素位置的象素值得到。

– The samples at half sample positions labelled b shall be derived by first calculating intermediate values denoted as b_1 by applying the 6-tap filter to the nearest integer position samples in the horizontal direction. The samples at half sample positions labelled h shall be derived by first calculating intermediate values denoted as h_1 by applying the 6-tap filter to the nearest integer position samples in the vertical direction:

– 标识为 b 的半象素位置的象素值是由标识为 b_1的中间值得到的，b_1 是对水平方向最近整象素位置的象素进行 6 阶滤波的结果。标识为 h 的半象素位置的象素值是由标识为 h_1 的中间值得到的，h_1 是对垂直方向最近整象素位置的象素进行 6 阶滤波的结果。

$$b_1 = (E - 5 * F + 20 * G + 20 * H - 5 * I + J) \quad (8\text{-}185)$$
$$h_1 = (A - 5 * C + 20 * G + 20 * M - 5 * R + T) \quad (8\text{-}186)$$

The final prediction values b and h shall be derived using:

最终预测值 b 和 h 的计算如下：

$$b = \text{Clip1}((b_1 + 16) >> 5) \quad (8\text{-}187)$$
$$h = \text{Clip1}((h_1 + 16) >> 5) \quad (8\text{-}188)$$

– The samples at half sample position labelled as j shall be derived by first calculating intermediate value denoted as j_1 by applying the 6-tap filter to the intermediate values of the closest half sample positions in either the horizontal or vertical direction because these yield an equal result.

– 标识为 j 的半象素坐标的象素值由标识为 j_1 的中间值计算得到，j_1 是对水平方向或垂直方向最近的半象素位置的象素值进行 6 阶滤波得到，水平方向和垂直方向滤波产生相同的结果。

$$j_1 = cc - 5 * dd + 20 * h_1 + 20 * m_1 - 5 * ee + ff, \text{ or} \quad (8\text{-}189)$$
$$j_1 = aa - 5 * bb + 20 * b_1 + 20 * s_1 - 5 * gg + hh \quad (8\text{-}190)$$

where intermediate values denoted as aa, bb, gg, s_1 and hh shall be derived by applying the 6-tap filter horizontally in the same manner as the derivation of b_1 and intermediate values denoted as cc, dd, ee, m_1 and ff shall be derived by applying the 6-tap filter vertically in the same manner as the derivation of h_1. The final prediction value j shall be derived using:

这里标识为 aa, bb, gg, s_1 和 hh的中间值是通过使用计算 b_1 相同的方式进行 6 阶滤波得到的，而标识为 cc, dd, ee, m_1 和 ff 的中间值是通过使用计算 h_1 相同的方式进行 6 阶滤波得到的，最终的预测值 j 的计算如下：

$$j = \mathrm{Clip1}((j_1 + 512) >> 10) \quad (8\text{-}191)$$

– The final prediction values s and m shall be derived from s_1 and m_1 in the same manner as the derivation of b and h, as given by:

– 最终的预测值 s 和 m 是按计算 b 和 h的方式由 s_1 和 m_1计算得到的。

$$s = \mathrm{Clip1}((s_1 + 16) >> 5) \quad (8\text{-}192)$$
$$m = \mathrm{Clip1}((m_1 + 16) >> 5) \quad (8\text{-}193)$$

– The samples at quarter sample positions labelled as a, c, d, n, f, i, k, and q shall be derived by averaging with upward rounding of the two nearest samples at integer and half sample positions using:

– 标识为 a, c, d, n, f, i, k和 q 的1/4象素位置的象素值是平均最近的半象素位置和整象素位置的象素并向上取整得到的。

$$a = (G + b + 1) >> 1 \quad (8\text{-}194)$$
$$c = (H + b + 1) >> 1 \quad (8\text{-}195)$$
$$d = (G + h + 1) >> 1 \quad (8\text{-}196)$$
$$n = (M + h + 1) >> 1 \quad (8\text{-}197)$$
$$f = (b + j + 1) >> 1 \quad (8\text{-}198)$$
$$i = (h + j + 1) >> 1 \quad (8\text{-}199)$$
$$k = (j + m + 1) >> 1 \quad (8\text{-}200)$$
$$q = (j + s + 1) >> 1. \quad (8\text{-}201)$$

– The samples at quarter sample positions labelled as e, g, p, and r shall be derived by averaging with upward rounding of the two nearest samples at half sample positions in the diagonal direction using

– 标识为 e, g, p, 和 r 的1/4象素位置的象素值是按对角方向平均两个最近的半象素位置的象素值并向上取整得到的。

$$e = (b + h + 1) >> 1 \quad (8\text{-}202)$$
$$g = (b + m + 1) >> 1 \quad (8\text{-}203)$$
$$p = (h + s + 1) >> 1 \quad (8\text{-}204)$$
$$r = (m + s + 1) >> 1. \quad (8\text{-}205)$$

The luma location offset in fractional-sample units ($xFrac_L$, $yFrac_L$) specifies which of the generated luma samples at full-sample and fractional-sample locations is assigned to the predicted luma sample value $predPartLX_L[x_L, y_L]$. This assignment is done according to Table 8-11. The value of $predPartLX_L[x_L, y_L]$ shall be the output.

分象素亮度位置的偏移 ($xFrac_L$, $yFrac_L$) 说明了哪一个在整象素位置和分象素位置产生的象素值将被赋给预测亮度象素值 $predPartLX_L[x_L, y_L]$，根据 Table 8-11 来完成这样的赋值过程，然后输出 $predPartLX_L[x_L, y_L]$ 值。

Table 8-11 – Assignment of the luma prediction sample $predPartLX_L[x_L, y_L]$

$xFrac_L$	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
$yFrac_L$	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
$predPartLX_L[x_L, y_L]$	G	d	h	n	a	e	i	p	b	f	j	q	c	g	k	r

8.4.2.2.2 Chroma sample interpolation process

8.4.2.2.2 色差象素的插值处理

Inputs to this process are

本过程的输入是：

- a chroma location in full-sample units ($xInt_C$, $yInt_C$),
- a chroma location offset in fractional-sample units ($xFrac_C$, $yFrac_C$), and

- chroma component samples from the selected reference picture $refPicLX_C$.
- 整象素的色差位置 ($xInt_C$, $yInt_C$)，
- 分象素色差位置的偏移($xFrac_C$, $yFrac_C$)，
- 选择的参考图像色差分量数组$refPicLX_C$

Output of this process is a predicted chroma sample value $predPartLX_C[x_C, y_C]$.

本过程的输出是预测色差象素值 $predPartLX_C[x_C, y_C]$。

In Figure 8-5, the positions labelled with A, B, C, and D represent chroma samples at full-sample locations inside the given two-dimensional array $refPicLX_C$ of chroma samples.

在 Figure 8-5 中，标识为A, B, C, 和 D 的位置表示在给定的二维色差象素数组 $refPicLX_C$ 中整象素色差象素。

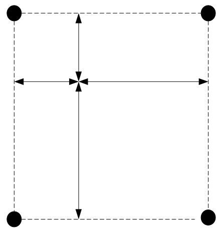


Figure 8-5 – Fractional sample position dependent variables in chroma interpolation and surrounding integer position samples A, B, C, and D.

These samples may be used for generating the predicted chroma sample value $predPartLX_C[x_C, y_C]$.

这些象素可能被用来产生预测色差象素值 $predPartLX_C[x_C, y_C]$。

$$xA_C = \text{Clip3}(0, PicWidthInSamples_C - 1, xInt_C) \quad (8\text{-}206)$$
$$xB_C = \text{Clip3}(0, PicWidthInSamples_C - 1, xInt_C + 1) \quad (8\text{-}207)$$
$$xC_C = \text{Clip3}(0, PicWidthInSamples_C - 1, xInt_C) \quad (8\text{-}208)$$
$$xD_C = \text{Clip3}(0, PicWidthInSamples_C - 1, xInt_C + 1) \quad (8\text{-}209)$$

$$yA_C = \text{Clip3}(0, PicHeightInSamples_C - 1, yInt_C) \quad (8\text{-}210)$$
$$yB_C = \text{Clip3}(0, PicHeightInSamples_C - 1, yInt_C) \quad (8\text{-}211)$$
$$yC_C = \text{Clip3}(0, PicHeightInSamples_C - 1, yInt_C + 1) \quad (8\text{-}212)$$
$$yD_C = \text{Clip3}(0, PicHeightInSamples_C - 1, yInt_C + 1) \quad (8\text{-}213)$$

Given the chroma samples A, B, C, and D at full-sample locations, the predicted chroma sample value $predPartLX_C[x_C, y_C]$ is derived as follows:

根据给定的整象素位置的色差象素 A, B, C, 和 D，预测色差象素值 $predPartLX_C[x_C, y_C]$ 的计算如下：

$$\begin{aligned} predPartLX_C[x_C, y_C] = ((8 - xFrac_C) * (8 - yFrac_C) * A + xFrac_C * (8 - yFrac_C) * B + \\ (8 - xFrac_C) * yFrac_C * C \quad + xFrac_C * yFrac_C * D \quad + 32) >> 6 \end{aligned} \quad (8\text{-}214)$$

8.4.2.3 Weighted sample prediction process

8.4.2.3 加权象素预测处理过程

Inputs to this process are

本过程的输入是：

- mbPartIdx: the current partition given by the partition index
- subMbPartIdx: the sub-macroblock partition index

- predFlagL0 and predFlagL1: prediction list utilization flags
- $predPartLX_L$: a (partWidth)x(partHeight) array of prediction luma samples (with LX being replaced by L0 or L1 depending on predFlagL0 and predFlagL1)
- $predPartLX_{Cb}$ and $predPartLX_{Cr}$: (partWidth/2)x(partHeight/2) arrays of prediction chroma samples, one for each of the chroma components Cb and Cr (with LX being replaced by L0 or L1 depending on predFlagL0 and predFlagL1)
- mbPartIdx：当前宏块部分的索引；
- subMbPartIdx：子宏块部分索引；
- predFlagL0 和 predFlagL1：预测列表使用标志；
- $predPartLX_L$：一 (partWidth)x(partHeight) 预测亮度象素数组（根据 predFlagL0 和 predFlagL1 的值，LX 可能为 L0 或 L1）；
- $predPartLX_{Cb}$ 和 $predPartLX_{Cr}$：(partWidth/2)x(partHeight/2) 预测色差象素数组（根据 predFlagL0 和 predFlagL1 的值，LX 可能为 L0 或 L1）

Outputs of this process are

本过程的输出是：

- $predPart_L$: a (partWidth)x(partHeight) array of prediction luma samples and
- $predPart_{Cb}$, and $predPart_{Cr}$: (partWidth/2)x(partHeight/2) arrays of prediction chroma samples, one for each of the chroma components Cb and Cr.
- $predPart_L$：一 (partWidth)x(partHeight) 预测亮度象素数组
- $predPart_{Cb}$ 和 $predPart_{Cr}$：(partWidth/2)x(partHeight/2) 预测色差象素数组，分别对应于两个色差分量Cb和Cr。

For macroblocks or partitions with predFlagL0 equal to 1 in P and SP slices, the following applies.

在 P 和 SP slice 中对于 predFlagL0 等于 1 的宏块或者宏块部分，使用下面的过程：

- If weighted_pred_flag is equal to 0, the default weighted sample prediction process as described in subclause 8.4.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause.
- Otherwise (weighted_pred_flag is equal to 1), the explicit weighted prediction process as described in subclause 8.4.2.3.2 is invoked with the same inputs and outputs as the process described in this subclause.
- 如果 weighted_pred_flag 等于 0，那么就使用与本小节处理过程相同的输入和输出参数，调用 8.4.2.3.1 小节中的缺省加权预测处理过程；
- 否则（weighted_pred_flag 等于 1），那么就使用与本小节处理过程相同的输入和输出参数，调用 8.4.2.3.2 小节中的显式加权预测过程；

For macroblocks or partitions with predFlagL0 or predFlagL1 equal to 1 in B slices, the following applies.

在 B slices 中对于 predFlagL0 或者 predFlagL1 等于 1 的宏块或宏块部分，使用下面的过程：

- If weighted_bipred_idc is equal to 0, the default weighted sample prediction process as described in subclause 8.4.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause.
- If weighted_bipred_idc is equal to 1, the explicit weighted sample prediction process as described in subclause 8.4.2.3.2, for macroblocks or partitions with predFlagL0 or predFlagL1 equal to 1 with the same inputs and outputs as the process described in this subclause.
- 如果 weighted_bipred_idc 等于 0，那么就使用与本小节处理过程相同的输入和输出参数，调用 8.4.2.3.1 小节中的缺省加权预测处理过程；
- 如果 weighted_bipred_idc 等于 1，对于 predFlagL0 或者 predFlagL1 等于 1 的宏块或者宏块部分来说，那么就使用与本小节处理过程相同的输入和输出参数，调用8.4.2.3.2 小节中的显式加权预测处理过程；
- Otherwise (weighted_bipred_idc is equal to 2), the following applies.
- 否则（weighted_bipred_idc is equal to 2），使用下面的处理过程。

- If predFlagL0 is equal to 1 and predFlagL1 is equal to 1, the implicit weighted sample prediction as described in subclause 8.4.2.3.2 is invoked with the same inputs and outputs as the process described in this subclause.
- Otherwise (predFlagL0 or predFlagL1 are equal to 1 but not both), the default weighted sample prediction process as described in subclause 8.4.2.3.1 is invoked with the same inputs and outputs as the process described in this subclause with the same inputs and outputs as the process described in this subclause.
- 如果 predFlagL0 和 predFlagL1 都等于 1，那么就使用与本小节处理过程相同的输入和输出参数，调用 8.4.2.3.2 小节中的隐式加权预测处理过程；
- 否则（predFlagL0 或 predFlagL1 等于 1，但不全等于1），那么就使用与本小节处理过程相同的输入和输出参数，调用 8.4.2.3.1 小节中的缺省加权预测处理过程；

8.4.2.3.1 Default weighted sample prediction process

8.4.2.3.1 缺省加权象素预测过程

Input to this process are the same as specified in subclause 8.4.2.3.

本过程的输入和 8.4.2.3 小节中的输入相同

Output of this process are the same as specified in subclause 8.4.2.3.

本过程的输出和8.4.2.3 小节中的输出相同

- If the luma sample prediction values $predPart_L[x, y]$ are derived, the following applies with C set equal to L, x set equal to 0 .. partWidth - 1, and y set equal to 0 .. partHeight - 1.
- If the chroma Cb component sample prediction values $predPart_{Cb}[x, y]$ are derived, the following applies with C set equal to Cb, x set equal to 0 .. partWidth / 2 - 1, and y set equal to 0 .. partHeight / 2 - 1.
- Otherwise (the chroma Cr component sample prediction values $predPart_{Cr}[x, y]$ are derived), the following applies with C set equal to Cb, x set equal to 0 .. partWidth / 2 - 1, and y set equal to 0 .. partHeight / 2 - 1.
- 如果是计算亮度象素预测值 $predPart_L[x, y]$，那么在下面的等式中 C 被设置成 L，x 被设置成 0 .. partWidth – 1，y 被设置成 0 .. partHeight – 1。
- 如果是计算色差分量 Cb 象素预测值 $predPart_{Cb}[x, y]$，那么在下面的等式中 C 被设置成 Cb, x 被设置成 0 .. partWidth / 2 - 1, y 被设置成 0 .. partHeight / 2 – 1 。
- 否则（计算色差分量 Cr 象素预测值 $predPart_{Cr}[x, y]$），那么在下面的等式中 C 被设置成 Cr, x 被设置成 0 .. partWidth / 2 - 1, y 被设置成 0 .. partHeight / 2 – 1 。

The prediction sample values are derived by

预测象素值的计算如下：

- If predFlagL0 is equal to 1 and predFlagL1 is equal to 0 for the current partition
- 如果对于当前部分来说，predFlagL0 等于 1 并且 predFlagL1 等于 0，那么

$$predPart_C[x, y] = predPartL0_C[x, y] \tag{8-215}$$

- If predFlagL0 is equal to 0 and predFlagL1 is equal to 1 for the current partition
- 如果对于当前部分来说，predFlagL0 等于 0 并且 predFlagL1 等于 1，那么

$$predPart_C[x, y]= predPartL1_C[x, y] \tag{8-216}$$

- Otherwise (predFlagL0 and predFlagL1 are equal to 1 for the current partition),
- 否则（对于当前部分来说，predFlagL0 和 predFlagL1 都等于 1）

$$predPart_C[x, y] = (predPartL0_C[x, y] + predPartL1_C[x, y] + 1) >> 1. \tag{8-217}$$

8.4.2.3.2 Weighted sample prediction process

8.4.2.3.2 加权象素预测处理

Input to this process are the same as specified in subclause 8.4.2.3.

本过程的输入和 8.4.2.3 小节中的输入相同

Output of this process are the same as specified in subclause 8.4.2.3.

本过程的输出和8.4.2.3 小节中的输出相同

- If the luma sample prediction values $\text{predPart}_L[\, x, y \,]$ are derived, the following applies with C set equal to L, x set equal to 0 .. partWidth - 1, and y set equal to 0 .. partHeight - 1.
- If the chroma Cb component sample prediction values $\text{predPart}_{Cb}[\, x, y \,]$ are derived, the following applies with C set equal to Cb, x set equal to 0 .. partWidth / 2 - 1, and y set equal to 0 .. partHeight / 2 - 1.
- Otherwise (the chroma Cr component sample prediction values $\text{predPart}_{Cr}[\, x, y \,]$ are derived), the following applies with C set equal to Cb, x set equal to 0 .. partWidth / 2 - 1, and y set equal to 0 .. partHeight / 2 - 1.
- 如果是计算亮度象素预测值 $\text{predPart}_L[\, x, y \,]$，那么在下面的等式中 C 被设置成 L，x 被设置成 0 .. partWidth – 1，y 被设置成 0 .. partHeight – 1。
- 如果是计算色差分量 Cb 象素预测值 $\text{predPart}_{Cb}[\, x, y \,]$，那么在下面的等式中 C 被设置成 Cb, x 被设置成 0 .. partWidth / 2 - 1, y 被设置成 0 .. partHeight / 2 – 1 。
- 否则（计算色差分量 Cr 象素预测值 $\text{predPart}_{Cr}[\, x, y \,]$），那么在下面的等式中 C 被设置成 Cr, x 被设置成 0 .. partWidth / 2 - 1, y 被设置成 0 .. partHeight / 2 – 1 。

The prediction sample values are derived by

预测象素值的计算如下：

- If the partition mbPartIdx\subMbPartIdx has predFlagL0 equal to 1 and predFlagL1 equal to 0, the final predicted sample values $\text{predPart}_C[\, x, y \,]$ are derived by
- 如果对于 mbPartIdx\subMbPartIdx 来说，predFlagL0 等于 1 并且 predFlagL1 等于 0 ，那么最终的预测值 $\text{predPart}_C[\, x, y \,]$ 的计算如下：

$$\begin{aligned}&\text{if(logWD >= 1)}\\&\quad \text{predPart}_C[\, x, y \,] = \text{Clip1}(\, (\, (\, \text{predPartL0}_C[\, x, y \,] * w_0 + 2^{\text{logWD - 1}}\,) >> \text{logWD}\,) + o_0\,)\\&\text{else}\\&\quad \text{predPart}_C[\, x, y \,] = \text{Clip1}(\, \text{predPartL0}_C[\, x, y \,] * w_0 + o_0\,)\end{aligned} \tag{8-218}$$

- If the partition mbPartIdx\subMbPartIdx has predFlagL0 equal to 0 and predFlagL1 equal to 1, the final predicted sample values $\text{predPart}_C[\, x, y \,]$ are derived by
- 如果对于 mbPartIdx\subMbPartIdx 来说，predFlagL0 等于 0 并且 predFlagL1 等于 1 ，那么最终的预测值 $\text{predPart}_C[\, x, y \,]$ 的计算如下：

$$\begin{aligned}&\text{if(logWD >= 1)}\\&\quad \text{predPart}_C[\, x, y \,] = \text{Clip1}(\, (\, (\, \text{predPartL1}_C[\, x, y \,] * w_1 + 2^{\text{logWD - 1}}\,) >> \text{logWD}\,) + o_1\,)\\&\text{else}\\&\quad \text{predPart}_C[\, x, y \,] = \text{Clip1}(\, \text{predPartL1}_C[\, x, y \,] * w_1 + o_1\,)\end{aligned} \tag{8-219}$$

- Otherwise (the partition mbPartIdx\subMbPartIdx has both predFlagL0 and predFlagL1 equal to 1), the final predicted sample values $\text{predPart}_C[\, x, y \,]$ are derived by
- 如果对于 mbPartIdx\subMbPartIdx 来说，predFlagL0 和 predFlagL1 都等于 1 ，那么最终的预测值 $\text{predPart}_C[\, x, y \,]$ 的计算如下：

$$\begin{aligned}\text{predPart}_C[\, x, y \,] = \text{Clip1}(\, (\, (\, &\text{predPartL0}_C[\, x, y \,] * w_0 + \text{predPartL1}_C[\, x, y \,] * w_1 + 2^{\text{logWD}}\,) >>\\&(\, \text{logWD} + 1\,)\,) + (\, (\, o_0 + o_1 + 1\,) >> 1\,)\,)\end{aligned} \tag{8-220}$$

The variables in the above derivation for the prediction samples are derived as follows.

在上述计算预测象素值中使用到的变量的计算如下：

- If weighted_bipred_idc is equal to 2 and the slice_type is equal to B,
- 如果 weighted_bipred_idc is equal to 2 等于 2 并且 slice_type 等于 B，那么（即隐式加权预测）

$$\text{logWD} = 5 \tag{8-221}$$

$o_0 = 0$ (8-222)

$o_1 = 0$ (8-223)

$w_1 = \text{DistScaleFactor} >> 2$ (8-224)

where DistScaleFactor is specified in subclause 8.4.1.2.3, and

这里 DistScaleFactor 的说明如 8.4.1.2.3 小节所示

$w_0 = 64 - w_1$ (8-225)

- If DiffPicOrderCnt(picA, picB) is equal to 0 with picA being the picture referred by RefPicList1[refIdxL1] and picB being the picture referred by RefPicList0[refIdxL0] or one or both reference pictures is a long-term reference picture or (DistScaleFactor >> 2) < -64 or (DistScaleFactor >> 2) > 128 where DistScaleFactor is specified in subclause 8.4.1.2.3
- 如果 DiffPicOrderCnt(picA, picB) 等于 0，这里 picA 为RefPicList1[refIdxL1] 表示的参考图像， picB 为 RefPicList0[refIdxL0] 表示的参考图像，或者这两参考图像中的一个或两个都是长期参考图像，或者 (DistScaleFactor >> 2) < -64 或 (DistScaleFactor >> 2) > 128 时，这里 DistScaleFactor 的说明如 8.4.1.2.3 小节所示，那么：

  $w_0 = 32$ (8-226)

  $w_1 = 32$ (8-227)

- Otherwise
- 否则

  $w_0 = 64 - (\text{DistScaleFactor} >> 2)$ (8-228)

  $w_1 = \text{DistScaleFactor} >> 2$ (8-229)

- Otherwise (weighted_pred_flag is equal to 1 in P slices or weighted_bipred_idc equal to 1 in B slices), explicit mode weighted prediction is used as follows.
- 否则（在 P slice 中 weighted_pred_flag 等于 1 或者在 B slice 中weighted_bipred_idc 等于 1），那么就使用显式的加权预测模式。
  - The variables refIdxL0WP and refIdxL1WP are derived as follows.
  - 变量 refIdxL0WP 和 refIdxL1WP 的计算如下：
    - If MbaffFrameFlag is equal to 1 and the current macroblock is a field macroblock
    - 如果 MbaffFrameFlag 等于 1，并且当前宏块是场宏块，那么

      refIdxL0WP = refIdxL0 >> 1 (8-230)

      refIdxL1WP = refIdxL1 >> 1 (8-231)

    - Otherwise (MbaffFrameFlag is equal to 0 or the current macroblock is a frame macroblock),
    - 否则（MbaffFrameFlag 等于 0，并且当前宏块是帧宏块），那么

      refIdxL0WP = refIdxL0 (8-232)

      refIdxL1WP = refIdxL1 (8-233)

  - The variables logWD, w_0, w_1, o_0, and o_1 are derived as follows.
  - 变量 logWD, w_0, w_1, o_0, 和 o_1 的计算如下：
    - If C in predPart_C[x, y] is replaced by L for luma samples
    - 对于亮度象素，predPart_C[x, y] 中的 C 由 L 取代

$\mathrm{logWD} = \mathrm{luma\_log2\_weight\_denom}$ (8-234)

$w_0 = \mathrm{luma\_weight\_l0[\ refIdxL0WP\ ]}$ (8-235)

$w_1 = \mathrm{luma\_weight\_l1[\ refIdxL1WP\ ]}$ (8-236)

$o_0 = \mathrm{luma\_offset\_l0[\ refIdxL0WP\ ]}$ (8-237)

$o_1 = \mathrm{luma\_offset\_l1[\ refIdxL1WP\ ]}$ (8-238)

- Otherwise (C in $\mathrm{predPart}_C$[x, y] is replaced by Cb or Cr for chroma samples, with iCbCr = 0 for Cb, iCbCr = 1 for Cr),
- 否则（$\mathrm{predPart}_C$[x, y] 中的 C 由Cb 或 Cr 取代，对于 Cb，iCbCr = 0 ，对于 Cr，iCbCr = 1）

$\mathrm{logWD} = \mathrm{chroma\_log2\_weight\_denom}$ (8-239)

$w_0 = \mathrm{chroma\_weight\_l0[\ refIdxL0WP\ ][\ iCbCr\ ]}$ (8-240)

$w_1 = \mathrm{chroma\_weight\_l1[\ refIdxL1WP\ ][\ iCbCr\ ]}$ (8-241)

$o_0 = \mathrm{chroma\_offset\_l0[\ refIdxL0WP\ ][\ iCbCr\ ]}$ (8-242)

$o_1 = \mathrm{chroma\_offset\_l1[\ refIdxL1WP\ ][\ iCbCr\ ]}$ (8-243)

When in explicit mode weighted prediction mode and predFlagL0 equal to 1 and predFlagL1 equal to 1, the following constraints shall be obeyed

在显式加权预测模式中并且 predFlagL0 和 predFlagL1 都等于 1，那么就必须遵循下面的限制：

$-128 <= w_0 + w_1 <= 127$ (8-244)

NOTE – For implicit mode weighted prediction, weights are guaranteed to be in the range is $-64 <= w_0, w_1 <= 128$.
注 一 对于显式加权预测模式，权的范围必须是 $-64 <= w_0, w_1 <= 128$。

8.5 Transform coefficient decoding process and picture construction process prior to deblocking filter process

8.5 环路滤波之前变换系数的解码过程和图像的重建过程

Inputs to this process are Intra16x16DCLevel (if available), Intra16x16ACLevel (if available), LumaLevel (if available), ChromaDCLevel, ChromaACLevel, and available Inter or Intra prediction sample arrays for the current macroblock for the applicable component pred_L, pred_{Cb}, or pred_{Cr}.

本过程的输入是 Intra16x16DCLevel （如果可用），Intra16x16ACLevel（如果可用），LumaLevel（如果可用），ChromaDCLevel, ChromaACLevel, 和可用的当前宏块的帧内或帧间预测象素数组 pred_L, pred_{Cb}, 或者 pred_{Cr}。

NOTE – When decoding a macroblock in Intra_4x4 prediction mode, the luma component of the macroblock prediction array may not be complete, since for each 4x4 luma block, the Intra_4x4 prediction process for luma samples as specified in subclause 8.3.1 and the process specified in this subclause are iterated.
注 一 当解码 Intra_4x4 预测模式的宏块时，宏块亮度分量的预测数组也许是不完全的，因为对于每一个 4x4 的亮度块来说，8.3.1 小节中的亮度分量 Intra_4x4 预测过程和本小节中的过程可能是重叠的。（具体说明哪部分是重叠的）

Outputs of this process are the constructed samples arrays prior to deblocking for the applicable component S'_L, S'_{Cb}, or S'_{Cr}.

本过程的输出是环路滤波之前可用的重建象素数组 S'_L, S'_{Cb}, 或 S'_{Cr}。

NOTE – When decoding a macroblock in Intra_4x4 prediction mode, the luma component of the macroblock constructed samples arrays prior to deblocking may not be complete, since for each 4x4 luma block, the Intra_4x4 prediction process for luma samples as specified in subclause 8.3.1 and the process specified in this subclause are iterated.
注 一 当解码 Intra_4x4 预测模式的宏块时，环路滤波之前的重建亮度分量数组可能是不完全的，因为对于每一个 4x4 的亮度块来说，8.3.1 小节中的亮度分量 Intra_4x4 预测过程和本小节中的过程可能是重叠的。（具体说明哪部分是重叠的）

This subclause specifies transform coefficient decoding and picture construction prior to the deblocking filter process.

本小节说明了环路滤波之前变换系数解码和图像重建的过程。

When the current macroblock is coded as P_Skip or B_Skip, all values of LumaLevel, ChromaDCLevel, ChromaACLevel are set equal to 0 for the current macroblock.

当当前宏块的编码模式是 P_Skip 或者 B_Skip ，那么当前宏块的LumaLevel, ChromaDCLevel, ChromaACLevel将被全部设置成 0。

8.5.1 Specification of transform decoding process for residual blocks

8.5.1 残差块变换解码过程的说明

When the current macroblock prediction mode is not equal to Intra_16x16, the variable LumaLevel contains the levels for the luma transform coefficients. For a 4x4 luma block indexed by luma4x4BlkIdx = 0..15, the following ordered steps are specified.

当当前宏块的预测模式不等于 Intra_16x16 时，变量 LumaLevel 包含了亮度变换系数值。对于索引为 luma4x4BlkIdx = 0..15 的 4x4 亮度块，依次使用下面的步骤：

1. The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with LumaLevel[luma4x4BlkIdx] as the input and the two-dimensional array c as the output.

   将 LumaLevel[luma4x4BlkIdx] 作为输入，使用 8.5.4 小节的反向变换系数扫描过程，输出一二维数组 c。

2. The scaling and transformation process for residual luma 4x4 blocks as specified in subclause 8.5.8 is invoked with c as the input and x’’ as the output.

   将 c 作为输入，使用 8.5.8 小节中的残差亮度 4x4 量化和变换处理过程并且输出 x’’。

3. The position of the upper-left sample of a 4x4 luma block with index luma4x4BlkIdx inside the macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (xO, yO).

   将 luma4x4BlkIdx 作为输入，使用 6.4.3 小节中的反向 4x4 亮度块扫描过程，计算出索引为 luma4x4BlkIdx 的 4x4 亮度块的左上角象素在当前宏块中的坐标位置 (xO, yO)。

4. The 4x4 array u with elements u_{yx} for x, y = 0..3 is derived as

   元素为u_{yx}（x, y = 0..3）的 4x4 数组的计算如下：

   $$u_{yx} = \mathrm{Clip1}(\mathrm{pred}_L[xO + x, yO + y] + x''_{yx}) \quad (8\text{-}245)$$

5. The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with luma4x4BlkIdx, u as the input and S’ as the output.

   将 luma4x4BlkIdx, u 作为输入，使用 8.5.9 小节中的环路滤波之前图像的重建过程，并且输出 S’。

8.5.2 Specification of transform decoding process for luma samples of Intra_16x16 macroblock prediction mode

8.5.2 Intra_16x16 宏块预测模式的亮度象素变换解码过程的说明

When the current macroblock prediction mode is equal to Intra_16x16, the variables Intra16x16DCLevel and Intra16x16ACLevel contain the levels for the luma transform coefficients. The transform coefficient decoding proceeds in the following ordered steps:

当当前宏块的预测模式等于 Intra_16x16 时，变量 Intra16x16DCLevel 和 Intra16x16ACLevel 包含了亮度变换系数值，这些亮度系数的解码依次使用下面的步骤。

1. The 4x4 luma DC transform coefficients of all 4x4 luma blocks of the macroblock are decoded.

   当前宏块的所有 4x4 亮度块的 DC 变换系数的解码。

   a. The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with Intra16x16DCLevel as the input and the two-dimensional array c as the output.

      将 Intra16x16DCLevel 作为输入，使用 8.5.4 小节中的反向变换系数扫描过程，输出一个二维数组 C。

   b. The scaling and transformation process for luma DC transform coefficients for Intra_16x16 macroblock type as specified in subclause 8.5.6 is invoked with c as the input and dcY as the output.

将 C 作为输入，使用 8.5.6 小节中的 Intra_16x16 宏块类型的亮度DC 变换系数的量化和变换过程并且输出 dcY。

2. For a 4x4 luma block indexed by luma4x4BlkIdx = 0..15, the following ordered steps are specified.

   对于 索引为 luma4x4BlkIdx = 0..15 的 4x4 亮度块，依次使用下面的步骤

   a. The variable lumaList, which is a list of 16 entries, is derived. The first entry of lumaList is the corresponding value from the array dcY. Figure 8-6 shows the assignment of the indices of the array dcY to the luma4x4BlkIdx. The two numbers in the small squares refer to indices i and j in dcY_{ij}, and the numbers in large squares refer to luma4x4BlkIdx.

   16 个元素列表 lumaList 的计算，lumaList 中的第一个元素是数组 dcY 中的对应值，Figure 8-6 显示了数组 dcY 到 luma4x4BlkIdx 的分配过程。小正方形中的两个数字表示 dcY_{ij} 中的索引 i 和 j，大正方形中的数字表示 luma4x4BlkIdx。

00 0	01 1	02 4	03 5
10 2	11 3	12 6	13 7
20 8	21 9	22 12	23 13
30 10	31 11	32 14	33 15

Figure 8-6 – Assignment of the indices of dcY to luma4x4BlkIdx.

   The elements in lumaList with index k = 1..15 are specified as

   在 lumaList 中索引 k = 1..15 的元素说明如下：

lumaList[k] = Intra16x16ACLevel[luma4x4BlkIdx][k - 1] (8-246)

   b. The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with lumaList as the input and the two-dimensional array c as the output.

   将 lumaList 作为输入，使用 8.5.4 小节中的反向变换系数扫描过程，并且输出二维数组 C。

   c. The scaling and transformation process for residual luma 4x4 blocks as specified in subclause 8.5.8 is invoked with c as the input and x’’ as the output.

   将 c 作为输入，使用 8.5.8 小节中的残差亮度 4x4 块的量化和变换处理过程并输出 x’’。

   d. The position of the upper-left sample of a 4x4 luma block with index luma4x4BlkIdx inside the macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (xO, yO).

   将 luma4x4BlkIdx 作为输入，使用 6.4.3 小节中的反向 4x4 亮度块扫描过程，计算出索引为 luma4x4BlkIdx 的 4x4 亮度块的左上角象素在当前宏块中的坐标位置 (xO, yO)。

   e. The 4x4 array u with elements u_{yx} for x, y = 0..3 is derived as

   元素为u_{yx}（x, y = 0..3）的 4x4 数组的计算如下：

$$u_{yx} = \mathrm{Clip1}(\mathrm{pred}_L[xO + x, yO + y] + x''_{yx}) \quad (8\text{-}247)$$

   f. The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with luma4x4BlkIdx, u as the input and S’ as the output.

   将 luma4x4BlkIdx, u 作为输入，使用 8.5.9 小节中的环路滤波之前图像的重建过程，并且输出 S’。

8.5.3 Specification of transform decoding process for chroma samples

8.5.3 色差象素变换解码过程的说明

For each chroma component, the variables ChromaDCLevel[iCbCr] and ChromaACLevel[iCbCr], with iCbCr set equal to 0 for Cb and iCbCr set equal to 1 for Cr, contain the levels for both components of the chroma transform coefficients. For each chroma component, the transform decoding proceeds separately in the following ordered steps:

对于每一个色差分量，变量 ChromaDCLevel[iCbCr] 和 ChromaACLevel[iCbCr] 包含了两个色差分量的变换系数值，其中对于 Cb 分量，iCbCr 被设置成 0，而对于 Cr 分量，iCbCr 被设置成 1。对于每一个色差分量，变换解码过程依次使用下面的步骤：

1. The 2x2 luma DC transform coefficients of the 4x4 chroma blocks of the component indexed by iCbCr of the macroblock are decoded.

   2x2 色差 DC 变换系数的解码，这些系数是当前宏块中索引为 iCbCr 的色差分量的 4x4 色差块的变换DC系数。

   a. The 2x2 array c is derived using the inverse raster scanning process applied to ChromaDCLevel as follows

      对 ChromaDCLevel 进行反向栅格扫描处理得到 2x2 数组 C。

$$c = \begin{bmatrix} \text{ChromaDCLevel}[\,\text{iCbCr}\,][\,0\,] & \text{ChromaDCLevel}[\,\text{iCbCr}\,][\,1\,] \\ \text{ChromaDCLevel}[\,\text{iCbCr}\,][\,2\,] & \text{ChromaDCLevel}[\,\text{iCbCr}\,][\,3\,] \end{bmatrix} \tag{8-248}$$

   b. The scaling and transformation process for chroma DC transform coefficients as specified in subclause 8.5.7 is invoked with c as the input and dcC as the output.

      将 C 作为输入，使用 8.5.7 小节中的色差 DC变换系数的量化和变换处理过程并且输出 dcC。

2. For each 4x4 chroma block indexed by chroma4x4BlkIdx = 0..3 of the component indexed by iCbCr, the following ordered steps are specified.

   对于色差分量 iCbCr索引为 chroma4x4BlkIdx = 0..3 的每一个 4x4 色差块，依次使用下面的步骤。

   a. The variable chromaList, which is a list of 16 entries, is derived. The first entry of chromaList is the corresponding value from the array dcC. Figure 8-7 shows the assignment of the indices of the array dcC to the chroma4x4BlkIdx. The two numbers in the small squares refer to indices i and j in dcC_{ij}, and the numbers in large squares refer to chroma4x4BlkIdx.

      16 个元素列表 chromaList 的计算，chromaList 中的第一个元素是数组 dcC中的对应值，Figure 8-7 显示了数组 dcC 到 chroma4x4BlkIdx的分配过程。小正方形中的两个数字表示 dcC_{ij} 中的索引 i 和 j，大正方形中的数字表示 chroma4x4BlkIdx。

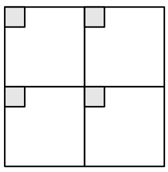


Figure 8-7 – Assignment of the indices of dcC to chroma4x4BlkIdx.

      The elements in chromaList with index k = 1..15 are specified as

      在 chromaList 中索引 k = 1..15 的元素说明如下：

$$\text{chromaList}[\,k\,] = \text{ChromaACLevel}[\,\text{chroma4x4BlkIdx}\,][\,k-1\,] \tag{8-249}$$

   b. The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with chromaList as the input and the two-dimensional array c as the output.

      将 chromaList 作为输入，使用 8.5.4 小节中的反向变换系数扫描过程，并且输出二维数组 C。

c. The scaling and transformation process for residual 4x4 blocks as specified in subclause 8.5.8 is invoked with c as the input and x’’ as the output.

将 c 作为输入，使用 8.5.8 小节中的残差亮度 4x4 块的量化和变换处理过程并输出 x’’。

d. The position of the upper-left sample of a 4x4 chroma block with index chroma4x4BlkIdx inside the macroblock is derived as follows

索引为 chroma4x4BlkIdx 的 4x4 色差块的左上角象素在当前宏块中的位置的计算如下：

$$xO = \text{InverseRasterScan}(\ \text{chroma4x4BlkIdx}, 4, 4, 8, 0\ ) \quad (8\text{-}250)$$

$$yO = \text{InverseRasterScan}(\ \text{chroma4x4BlkIdx}, 4, 4, 8, 1\ ) \quad (8\text{-}251)$$

e. The 4x4 array u with elements u_{yx} for x, y = 0..3 is derived as

元素为u_{yx}（x, y = 0..3）的 4x4 数组的计算如下：

$$u_{yx} = \text{Clip1}(\ \text{pred}_C[\ xO + x, yO + y\ ] + x''_{yx}\ ) \quad (8\text{-}252)$$

f. The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with chroma4x4BlkIdx, u as the input and S’ as the output.

将 luma4x4BlkIdx, u 作为输入，使用 8.5.9 小节中的环路滤波之前图像的重建过程，并且输出 S’。

8.5.4 Inverse scanning process for transform coefficients

8.5.4 变换系数的反向扫描过程

Input to this process is a list of 16 values.

本过程的输入是包含 16 个值的列表

Output of this process is a variable c containing a two-dimensional array of 4x4 values with level assigned to locations in the transform block.

本过程的输出是一变量 C，包含一个将系数排列在对应变换块位置的二维 4x4数组。

The decoding process maps the sequence of transform coefficient levels to the transform coefficient level positions. For this mapping, the two inverse scanning patterns shown in Figure 8-8 are used.

解码过程将变换系数值序列映射到对应的变换系数值位置，为了这个映射，使用了 Figure 8-8 中的两种反向扫描方式。

The inverse zig-zag scan shall be used for frame macroblocks and the inverse field scan shall be used for field macroblocks.

反向 zig-zag 扫描应用于帧宏块，而反向场扫描应用于场宏块。

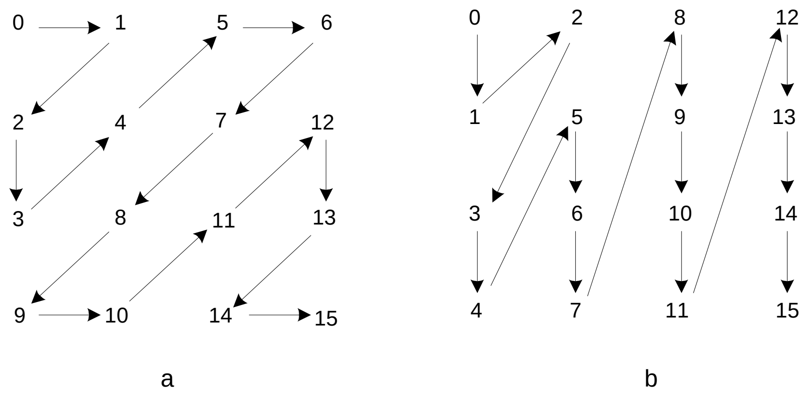


Figure 8-8 – a) Zig-zag scan. b) Field scan

Table 8-12 provides the mapping from the index idx of input list of 16 elements to indices i and j of the two-dimensional array c.

Table 8-12 说明了从16个元素的输入列表的索引 idx 到二维的数组 c 的索引 i 和 j 的映射。

Table 8-12 – Specification of mapping of idx to c_{ij} for zig-zag and field scan

idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
zig-zag	c_{00}	c_{01}	c_{10}	c_{20}	c_{11}	c_{02}	c_{03}	c_{12}	c_{21}	c_{30}	c_{31}	c_{22}	c_{13}	c_{23}	c_{32}	c_{33}
field	c_{00}	c_{10}	c_{01}	c_{20}	c_{30}	c_{11}	c_{21}	c_{31}	c_{02}	c_{12}	c_{22}	c_{32}	c_{03}	c_{13}	c_{23}	c_{33}

8.5.5 Derivation process for the quantisation parameters and scaling function

8.5.5 量化参数和缩放函数的处理过程

Input to this process is a two-dimensional array of transform coefficient levels.

本过程的输入一变换系数值的二维数组。

Outputs of this process are:

本过程的输出是：

– QP_C: the chroma quantisation parameter

– QS_C: the additional chroma quantisation parameter required for decoding SP and SI slices (if applicable)

– QP_C: 色差量化参数

– QS_C: 解码 SP 和 SI slices 需要的额外量化参数（如果可用的话）

QP quantisation parameter values QP_Y, QP_C, QS_Y, and QS_C shall be in the range of 0 to 51, inclusive.

QP 量化参数值 QP_Y, QP_C, QS_Y, QS_C 应该在 0 到 51 的范围内，包括 0 和 51 。

The value of QP_C for chroma is determined from the current value of QP_Y and the value of chroma_qp_index_offset.

色差的量化值 QP_C是由 QP_Y和 chroma_qp_index_offset 来决定的。

NOTE – The scaling equations are specified such that the equivalent quantisation parameter doubles for every increment of 6 in QP_Y. Thus, there is an increase in the factor used for scaling of approximately 12 % for each increase of 1 in the value of QP_Y.

注 一 缩放函数在 QP_Y 每增加 6 时，那么量化参数翻一倍，因此 QP_Y每增加一个单位，那么量化参数增加 12%。

The value of QP_C shall be determined as specified in Table 8-13 based on the indexing denoted qP_I. The value of qP_I shall be derived as follows.

QP_C 的值根据索引 qP_I 由 Table 8-13 得到，qP_I 值的计算如下：

$$qP_I = \text{Clip3}(0, 51, QP_Y + \text{chroma\_qp\_index\_offset}) \quad (8\text{-}253)$$

Table 8-13 – Specification of QP_C as a function of qP_I

qP_I	<30	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
QP_C	$=QP_I$	29	30	31	32	32	33	34	34	35	35	36	36	37	37	37	38	38	38	39	39	39	39

When the current slice is an SP or SI slice, QS_C is derived using the above process, substituting QP_Y with QS_Y and QP_C with QS_C.

当当前的 slice 为 SP 或者 SI slice 时，那么利用上面的过程计算出 QS_C，用 QS_Y 替代 QP_Y ,用 QS_C 替代 QP_C。

The function LevelScale(m, i, j) is specified as follows:

函数 LevelScale(m, i, j) 的说明如下：

$$\text{LevelScale}(m,i,j) = \begin{cases} v_{m0} & \text{for} \quad (i,j) \in \{(0,0),(0,2),(2,0),(2,2)\}, \\ v_{m1} & \text{for} \quad (i,j) \in \{(1,1),(1,3),(3,1),(3,3)\}, \\ v_{m2} & \text{otherwise;} \end{cases} \tag{8-254}$$

where the first and second subscripts of v are row and column indices, respectively, of the matrix specified as:

这里 V 的第一个下标和第二个下标分别是下面矩阵行和列的索引。

$$v = \begin{bmatrix} 10 & 16 & 13 \\ 11 & 18 & 14 \\ 13 & 20 & 16 \\ 14 & 23 & 18 \\ 16 & 25 & 20 \\ 18 & 29 & 23 \end{bmatrix}. \tag{8-255}$$

8.5.6 Scaling and transformation process for luma DC transform coefficients for Intra_16x16 macroblock type

8.5.6 Intra_16x16 宏块类型的亮度DC变换系数的缩放和变换处理

Inputs to this process are transform coefficient level values for luma DC transform coefficients of Intra_16x16 macroblocks as a 4x4 array c of elements c_{ij}, where i and j form a two-dimensional frequency index.

本过程的输入是 Intra_16x16 宏块类型的亮度 DC变换系数的变换系数值，表示成元素为 c_{ij} 的 4x4 数组 C，这里的 i 和 j 是二维的频率索引。

Outputs of this process are 16 scaled DC values for luma 4x4 blocks of Intra_16x16 macroblocks as a 4x4 array dcY of elements dcY_{ij}.

本过程的输出是 16 个缩放的Intra_16x16 宏块的亮度 4x4 块的 DC 系数值，由元素为 dcY_{ij} 的 4x4 数组 dcY 表示。

The inverse transform for the 4x4 luma DC transform coefficients is specified by:

4x4 亮度 DC 变换系数的反变换过程的说明如下：

$$f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}. \tag{8-256}$$

A bitstream conforming to this Recommendation | International Standard shall not contain data that results in any element of f that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive.

遵循本协议的码流中不包含导致矩阵 f 中的元素超出整数值-2^{15} 到 $2^{15}-1$（包括-2^{15} 和 $2^{15}-1$）的数据。

After the inverse transform, scaling is performed according to the following:

反变换后，根据下面的步骤进行缩放操作：

- If QP_Y is greater than or equal to 12, the scaled result shall be derived as
- 如果 QP_Y 大于或等于 12 ，那么使用下面的方法计算缩放的结果

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0)) << (QP_Y / 6 - 2), \quad i, j = 0..3 . \tag{8-257}$$

- Otherwise (QP_Y is less than 12), the scaled results shall be derived as
- 否则（QP_Y 小于12），那么使用下面的方法计算缩放的结果

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0) + 2^{1-QP/6}) >> (2 - QP_Y / 6), \quad i, j = 0..3 . \tag{8-258}$$

A bitstream conforming to this Recommendation | International Standard shall not contain data that results in any element of dcY_{ij} that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive.

遵循本协议的码流中不包含导致矩阵 dcY_{ij} 中的元素超出整数值-2^{15} 到 $2^{15}-1$（包括-2^{15} 和 $2^{15}-1$）的数据。

NOTE – Care should be used in the design of encoders to avoid difficulty with meeting the dynamic range requirements of the decoding process for Intra_16x16 macroblocks when using small values of QP_Y (particularly for $QP_Y < 6$).

注 — 在设计编码器的过程中应该注意当使用较小的 QP_Y 值（特别$QP_Y < 6$）时对于Intra_16x16宏块解码过程的动态范围需求的困难。

8.5.7 Scaling and transformation process for chroma DC transform coefficients

8.5.7 色差 DC 变换系数的缩放和变换过程

Inputs to this process are transform coefficient level values for chroma DC transform coefficients of one chroma component of the macroblock as a 2x2 array c of elements c_{ij}, where i and j form a two-dimensional frequency index.

本过程的输入是当前宏块的一个色差分量的DC变换系数的变换值，表示成元素为 c_{ij} 的 2x2 数组 C，这里的 i 和 j 是二维的频率索引。

Outputs of this process are 4 scaled DC values as a 2x2 array dcC of elements dcC_{ij}.

本过程的输出是 4 个缩放 DC 系数值，由元素为 dcC_{ij}的2x2数组 dcC 表示。

The inverse transform for the 2x2 chroma DC transform coefficients is specified by:

2x2 色差 DC 变换系数的反变换说明如下：

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{8-259}$$

A bitstream conforming to this Recommendation | International Standard shall not contain data that results in any element f_{ij} of f that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive.

遵循本协议的码流中不包含导致矩阵f中的任何一个元素 f_{ij}超出整数值-2^{15} 到 $2^{15}-1$（包括-2^{15}和 $2^{15}-1$）的数据。

After the inverse transform, scaling is performed according to the following.

反变换后，根据下面的步骤进行缩放操作：

- If QP_C is greater than or equal to 6, the scaling result shall be derived as
- 如果 QP_C 大于或者等于 6，那么使用下面的方法计算缩放的结果

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP_C \% 6, 0, 0)) << (QP_C / 6 - 1), \quad i, j = 0, 1 \tag{8-260}$$

- Otherwise (QP_C is less than 6), the scaling results shall be derived by
- 否则（QP_C 小于 6），那么使用下面的方法计算缩放的结果

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP \% 6, 0, 0)) >> 1, \quad i, j = 0, 1 \tag{8-261}$$

A bitstream conforming to this Recommendation | International Standard shall not contain data that results in any element of dcC_{ij} that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive.

遵循本协议的码流中不包含导致矩阵dcC_{ij}中的任何一个元素超出整数值-2^{15}到 $2^{15}-1$（包括-2^{15} 和 $2^{15}-1$）的数据。

8.5.8 Scaling and transformation process for residual 4x4 blocks

8.5.8 残差 4x4 块的缩放和变换过程

Input to this process is a two-dimensional array c of elements c_{mn} which is either an array relating to a luma residual block or to a residual block of a chroma component.

本过程的输入是元素为 c_{mn} 的二维数组 C，它可能是一亮度残差块或者一色差分量的残差块。

Outputs of this process are residual sample values as 4x4 array x of elements x_{mn}.

本过程的输出是残差象素值，由元素为 x_{mn} 的 4x4 数组 x 表示。

The variable sMbFlag is derived as follows.

变量 sMbFlag 的计算如下：

- If mb_type is equal to SI or the macroblock prediction mode is equal to Inter in an SP slice, sMbFlag is set equal to 1,
- 在 SP slice 中，如果 mb_type 等于 SI 或者宏块的预测模式为帧间预测模式，那么 sMbFlag 将被设置成 1。
- Otherwise (mb_type not equal to SI and the macroblock prediction mode is not equal to Inter in an SP slice), sMbFlag is set equal to 0.
- 否则（在 SP slice 中，如果 mb_type 不等于 SI 并且宏块的预测模式也不为帧间预测模式），sMbFlag 将被设置成 1。

The variable qP is derived as follows.

变量 qP 的计算如下：

- If the input array c relates to a luma residual block and sMbFlag is equal to 0

  如果输入数组是一亮度残差块并且 sMbFlag 等于 0

$$qP = QP_Y \tag{8-262}$$

- If the input array c relates to a luma residual block and sMbFlag is equal to 1

  如果输入数组是一亮度残差块并且 sMbFlag 等于 1

$$qP = QS_Y \tag{8-263}$$

- If the input array c relates to a chroma residual block and sMbFlag is equal to 0

  如果输入数组是一色差残差块并且 sMbFlag 等于 0

$$qP = QP_C \tag{8-264}$$

- Otherwise (the input array c relates to a chroma residual block and sMbFlag is equal to 1)

  如果输入数组是一色差残差块并且 sMbFlag 等于 1

$$qP = QS_C \tag{8-265}$$

Scaling of 4x4 block transform coefficient levels c_{ij} proceeds as follows.

4x4 块变换系数值 c_{ij} 的缩放过程如下所示：

- If all of the following conditions are true

  如果所有下面的条件都正确

  - i is equal to 0

    i 等于 0

  - j is equal to 0

j 等于 0

- c relates to a luma residual block coded using Intra_16x16 prediction mode or c relates to a chroma residual block
- c 为使用 Intra_16x16 预测模式的亮度残差块或者 c 为色差残差块

$$w_{00} = c_{00} \tag{8-266}$$

- Otherwise
- 否则

$$w_{ij} = (c_{ij} * \mathrm{LevelScale}(qP \% 6, m, n)) << (qP / 6), \quad i, j = 0..3 \tag{8-267}$$

The bitstream shall not contain data that results in a value of w_{ij} that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive.

遵循本协议的码流中不包含导致 w_{ij} 的值超出整数值-2^{15}到 $2^{15}-1$（包括-2^{15} 和 $2^{15}-1$）的数据。

After constructing an entire 4x4 block of scaled transform coefficients and assembling these into a 4x4 array w of elements w_{ij} illustrated as

构造一个完整的缩放变换系数的4x4块并且将它们组合成元素为 w_{ij} 的一个 4x4 数组 W，如下所示。

$$w = \begin{bmatrix} w_{00} & w_{01} & w_{02} & w_{03} \\ w_{10} & w_{11} & w_{12} & w_{13} \\ w_{20} & w_{21} & w_{22} & w_{23} \\ w_{30} & w_{31} & w_{32} & w_{33} \end{bmatrix} \tag{8-268}$$

The transform process shall convert the block of scaled transform coefficients to a block of output samples in a manner mathematically equivalent to the following process:

变换过程是按下面的数学方式将缩放系数块转换成一个输出象素块。

1. First, each (vertical) column of scaled transform coefficients is transformed using a one-dimensional inverse transform, and

   首先，使用一维反向变换对缩放系数的列进行变换

2. Then, each (horizontal) row of the resulting matrix is transformed using the same one-dimensional inverse transform.

   然后，再使用一维反向变换对缩放系数的行进行变换

The one-dimensional inverse transform is specified as follows for four input samples w_0, w_1, w_2, w_3, where the subscript indicates the one-dimensional frequency index.

一维反向变换的说明如下，输入为 w_0, w_1, w_2, w_3,这里的下标表示一维频率索引。

1. A set of intermediate values is computed:

   中间数据集的计算：

$$z_0 = w_0 + w_2 \tag{8-269}$$

$$z_1 = w_0 - w_2 \tag{8-270}$$

$$z_2 = (w_1 >> 1) - w_3 \tag{8-271}$$

$$z_3 = w_1 + (w_3 >> 1) \tag{8-272}$$

2. The transformed result is computed from these intermediate values:

   根据中间数据值，计算出最终的变换值：

$$x_0 = z_0 + z_3 \tag{8-273}$$

$$x_1 = z_1 + z_2 \tag{8-274}$$

$$x_2 = z_1 - z_2 \tag{8-275}$$

$$x_3 = z_0 - z_3 \tag{8-276}$$

The bitstream shall not contain data that results in a value of z_0, z_1, z_2, z_3, x_0, x_1, x_2, or x_3 that exceeds the range of integer values from -2^{15} to $2^{15}-1$, inclusive, in either the first (vertical) or second (horizontal) stage of application of this transformation process. The bitstream shall not contain data that results in a value of x_0, x_1, x_2, or x_3 that exceeds the range of integer values from -2^{15} to $2^{15}-33$, inclusive, in the second (horizontal) stage of application of this transformation process.

不管在这个变换过程中的水平变换阶段还是垂直变换阶段，遵循本协议的码流中都不包含导致 z_0, z_1, z_2, z_3, x_0, x_1, x_2, 或 x_3 的值超出整数值-2^{15}到 $2^{15}-1$（包括-2^{15} 和 $2^{15}-1$）的数据。在这个变换的水平变换阶段，码流中也不包括导致 x_0, x_1, x_2, 或者 x_3 超出 -2^{15} 到 $2^{15}-33$范围的数据，包括 -2^{15} 和 $2^{15}-33$。

After performing both the one-dimensional vertical and the one-dimensional horizontal inverse transforms to produce a array of transformed samples,

执行完一维水平反变换和一位垂直反变换之后，将产生一个变换象素的数组，

$$x' = \begin{bmatrix} x'_{00} & x'_{01} & x'_{02} & x'_{03} \\ x'_{10} & x'_{11} & x'_{12} & x'_{13} \\ x'_{20} & x'_{21} & x'_{22} & x'_{23} \\ x'_{30} & x'_{31} & x'_{32} & x'_{33} \end{bmatrix}, \tag{8-277}$$

the final reconstructed sample residual values shall be derived as

最终的重建象素残差值的计算如下：

$$x''_{mn} = (x'_{mn} + 2^5) >> 6 \tag{8-278}$$

8.5.9 Picture construction process prior to deblocking filter process

8.5.9 环路滤波之前图像的重建过程

Inputs to this process are

本过程的输入是：

- luma4x4BlkIdx or chroma4x4BlkIdx
- a constructed sample residual 4x4 array x'' with elements x''_{yx} which is either a luma or chroma residual block
- the prediction sample 4x4 array $pred_L$, $pred_{Cb}$, $pred_{Cr}$
- luma4x4BlkIdx 或者 chroma4x4BlkIdx
- 重建象素残差的 4x4 数组 x''，它的元素 x''_{yx} 可能是亮度或者色差的残差块。
- 预测象素的 4x4 数组 $pred_L$, $pred_{Cb}$, $pred_{Cr}$

Outputs of this process are constructed sample blocks s' prior to the deblocking filter process.

本过程的输出是环路滤波之前的重建象素块 s'。

The position of the upper-left luma sample of the current macroblock is derived by invoking the inverse macroblock scanning process in subclause 6.4.1 with CurrMbAddr as input and the output being assigned to (xP, yP).

将 CurrMbAddr 作为输入，使用 6.4.1 小节中的反向宏块扫描过程，计算出当前宏块的左上角象素的坐标位置。

When x'' is a luma block, for each sample at position (x, y) of the 4x4 luma block, the following applies.

当 x'' 为一亮度块，对于 4x4 亮度块内坐标位置为 (x, y)的象素来说，使用下面的过程：

- The position of the upper-left sample of a 4x4 luma block with index luma4x4BlkIdx inside the macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (xO, yO).

  将 luma4x4BlkIdx 作为输入，使用 6.4.3 小节中的反向 4x4 亮度块扫描过程，计算出当前宏块中索引为 luma4x4BlkIdx 的4x4 亮度块的左上角象素的位置坐标 (xO, yO)，并且输出。

- If MbaffFrameFlag is equal to 1 and the current macroblock is a field macroblock

  如果 MbaffFrameFlag 等于 1，并且当前宏块是一场宏块

$$S'_L[\, xP + xO + x,\ yP + 2 * (\, yO + y\,)\,] = u_{yx} \tag{8-279}$$

- Otherwise (MbaffFrameFlag is equal to 0 or the current macroblock is a frame macroblock),

  如果 MbaffFrameFlag 等于 0，并且当前宏块是一帧宏块

$$S'_L[\, xP + xO + x,\ yP + yO + y\,] = u_{yx} \tag{8-280}$$

When x’’ is a chroma block, for each sample at position (x, y) of the 4x4 chroma block, the following applies.

当 x’’ 为一色差块，对于 4x4 色差块内坐标位置为 (x, y)的象素来说，使用下面的过程：

- The subscript C in the variables S'_C and $pred_C$ is replaced with Cb for the Cb chroma component and with Cr for the Cr chroma component.

  变量 S'_C 和 $pred_C$ 中的下表 C，对于Cb 色差分量由 Cb取代，而对于Cr 色差分量由 Cr 取代。

- The position of the upper-left sample of a 4x4 chroma block with index chroma4x4BlkIdx inside the macroblock is derived as follows

  当前宏块中索引为 chroma4x4BlkIdx 的 4x4色差块的左上角象素的坐标位置的计算如下所示：

$$xO = \text{InverseRasterScan}(\, \text{chroma4x4BlkIdx}, 4, 4, 8, 0\,) \tag{8-281}$$

$$yO = \text{InverseRasterScan}(\, \text{chroma4x4BlkIdx}, 4, 4, 8, 1\,) \tag{8-282}$$

- If MbaffFrameFlag is equal to 1 and the current macroblock is a field macroblock

  如果 MbaffFrameFlag 等于 1 并且 当前宏块是一场宏块，那么

$$S'_C[\, (\, xP >> 1\,) + xO + x,\ (\, (\, yP + 1\,) >> 1\,) + 2 * (\, yO + y\,)\,] = u_{yx} \tag{8-283}$$

- Otherwise (MbaffFrameFlag is equal to 0 or the current macroblock is a frame macroblock),

  否则（MbaffFrameFlag 等于 0 或者 当前宏块是一帧宏块）

$$S'_C[\, (\, xP >> 1\,) + xO + x,\ (\, (\, yP + 1\,) >> 1\,) + yO + y\,] = u_{yx} \tag{8-284}$$

8.6 Decoding process for P macroblocks in SP slices or SI macroblocks

8.6 SP slices 中P宏块和 SI 宏块的解码过程

This process is invoked when decoding P macroblock types in an SP slice type or an SI macroblock type in SI slices.

在 SP slice 中解码 P 宏块和在 SI slice 中解码 SI 宏块时使用本过程。

Inputs to this process are the prediction residual transform coefficient levels and the predicted samples for the current macroblock.

本过程的输入是：预测残差变换系数值和当前宏块的预测象素值。

Outputs of this process are the decoded samples of the current macroblock prior to deblocking.

本过程的输出是环路滤波之前的当前宏块的解码象素值。

This subclause specifies the transform coefficient decoding process and picture construction process for P macroblock types in SP slices and SI macroblock type in SI slices.

本小节说明了在 SP slice 中解码 P 宏块和在 SI slice 中解码 SI 宏块时变换系数的解码过程和图像的重建过程。

NOTE – SP slices make use of Inter predictive coding to exploit temporal redundancy in the sequence, in a similar manner to P slices. Unlike P slices, however, SP slice coding allows identical reconstruction of a slice even when different reference pictures are being used. SI slices make use of spatial prediction, in a similar manner to I slices. SI slice coding allows identical reconstruction to a corresponding SP slice. The properties of SP and SI slices aid in providing functionalities for bitstream switching, splicing, random access, fast-forward, fast reverse, and error resilience/recovery.

注 — 类似于 P slice，SP slices 使用帧间预测编码来去除序列之间的时间冗余。SP slice 保证了当使用不同的参考图像时重建出相同的 slice。类似于 I slice，SI slices 使用了帧内预测编码。SI slices 保证了和对应的 SP slice 有相同的重建。采用 SP 和 SI slices 的目的在于提供比特流的切换，拼接，随即访问，快进，快退，抗误和错误恢复等于功能。

An SP slice consists of macroblocks coded either as I macroblock types or P macroblock types.

SP slice 中的宏块可能是 I 宏块类型或者 P 宏块类型。

An SI slice consists of macroblocks coded either as I macroblock types or SI macroblock type.

SP slice 中的宏块可能是 I 宏块类型或者 SI 宏块类型。

The transform coefficient decoding process and picture construction process prior to deblocking filter process for I macroblock types in SI slices shall be invoked as specified in subclause 8.5. SI macroblock type shall be decoded as described below.

在 SI slice 中 I 类型宏块的环路滤波之前的变换系数的解码过程和图像的重建过程如 8.5 小节所示，SI 类型宏块的解码如下所示。

When the current macroblock is coded as P_Skip, all values of LumaLevel, ChromaDCLevel, ChromaACLevel are set equal to 0 for the current macroblock.

当当前宏块被编码成 P_Skip 类型时，那么当前宏块的 LumaLevel, ChromaDCLevel, ChromaACLevel所有值将被设置成0。

8.6.1 SP decoding process for non-switching pictures

8.6.1 非切换图像的 SP 解码过程

This process is invoked, when decoding P macroblock types in SP slices in which sp_for_switch_flag is equal to 0.

当解码 sp_for_switch_flag 等于 0 的 SP slice 中的 P 类型的宏块时使用本过程。

Input to this process are Inter prediction samples for the current macroblock from subclause 8.4 and the prediction residual transform coefficient levels.

本过程的输入是 8.4 小节中当前宏块的帧间预测象素和预测残差变换系数值。

Outputs of this process are the decoded samples of the current macroblock prior to the deblocking filter process.

本过程的输出是环路滤波之前的当前宏块的解码象素。

This subclause applies to all macroblocks in SP slices in which sp_for_switch_flag is equal to 0, except those with macroblock prediction mode equal to Intra_4x4 or Intra_16x16. It does not apply to SI slices.

本小节应用于 sp_for_switch_flag 等于 0 的 SP slice 中的所有宏块，除了预测模式等于 Intra_4x4 或 Intra_16x16的宏块，但它不应用于 SI slice。

8.6.1.1 Luma transform coefficient decoding process

8.6.1.1 亮度变换系数的解码过程

Inputs to this process are Inter prediction luma samples for the current macroblock $pred_L$ from subclause 8.4 and the prediction residual transform coefficient levels, LumaLevel, and the index of the 4x4 luma block luma4x4BlkIdx.

本过程的输入是 8.4 小节中的当前宏块的帧间预测亮度象素 $pred_L$ 和预测残差变换系数值 LumaLevel， 和 4x4 亮度块的索引 luma4x4BlkIdx。

Outputs of this process are the decoded luma samples of the current macroblock prior to the deblocking filter process.

本过程的输出是环路滤波之前当前宏块的解码象素值。

The position of the upper-left sample of the 4x4 luma block with index luma4x4BlkIdx inside the current macroblock is derived by invoking the inverse 4x4 luma block scanning process in subclause 6.4.3 with luma4x4BlkIdx as the input and the output being assigned to (x, y).

将 luma4x4BlkIdx 作为输入，使用 6.4.3 小节的反向 4x4 亮度块扫描处理过程，计算出索引为 luma4x4BlkIdx 的 4x4 亮度块的左上角象素在当前宏块中的坐标位置，并且输出 (x, y)。

Let the variable p be a 4x4 array of prediction samples with element p_{ij} being derived as follows.

元素为 p_{ij} 的 4x4 预测象素数组变量 p 的计算如下：

$$p_{ij} = \mathrm{pred}_L[\ y + j,\ x + i\ ] \quad (8\text{-}285)$$

The variable p is transformed producing transform coefficients c^p according to:

根据下面的等式对变量 p 进行变换产生变换系数 c^p 。

$$c^p = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \\ p_{30} & p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \quad (8\text{-}286)$$

The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with LumaLevel[luma4x4BlkIdx] as the input and the two-dimensional array c^r as the output with elements c_{ij}^r.

将LumaLevel[luma4x4BlkIdx]作为输入，使用 8.5.4 小节中的反向系数扫描过程，输出元素为 c_{ij}^r 的二维数组 c^r。

The prediction residual transform coefficients c^r are scaled using quantisation parameter QP_Y, and added to the transform coefficients of the prediction block c^p with i, j = 0..3 as follows.

使用量化系数 QP_Y 对预测残差变换系数 c^r 进行缩放处理并和预测块的变换系数 c^p 进行相加，i, j = 0..3，如下所示。

$$c_{ij}^s = c_{ij}^p + (\ (\ (\ c_{ij}^r * \mathrm{LevelScale}(\ QP_Y \,\%\, 6,\ i,\ j\ ) * A_{ij}\ ) << (\ QP_Y / 6\ )\ ) >> 6\ ) \quad (8\text{-}287)$$

where LevelScale(m, i, j) is specified in Equation 8-254, and where A_{ij} is specified as:

这里 LevelScale(m, i, j) 的说明如等式 8-254 所示，而 A_{ij} 的说明如下：

$$A_{ij} = \begin{cases} 16 & \text{for} \quad (i, j) \in \{(0,0), (0,2), (2,0), (2,2)\}, \\ 25 & \text{for} \quad (i, j) \in \{(1,1), (1,3), (3,1), (3,3)\}, \\ 20 & \text{otherwise;} \end{cases} \quad (8\text{-}288)$$

The function LevelScale2(m, i, j), used in the formulas below, is specified as:

下面等式中的函数 LevelScale2(m, i, j) 说明如下：

$$\mathrm{LevelScale2}(m, i, j) = \begin{cases} w_{m0} & \text{for} \quad (i, j) \in \{(0,0), (0,2), (2,0), (2,2)\}, \\ w_{m1} & \text{for} \quad (i, j) \in \{(1,1), (1,3), (3,1), (3,3)\}, \\ w_{m2} & \text{otherwise;} \end{cases} \quad (8\text{-}289)$$

where the first and second subscripts of w are row and column indices, respectively, of the matrix specified as:

w 的第一个下标和第二个下标分别为下面矩阵的行与列的索引。

$$w = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7490 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix} \quad (8\text{-}290)$$

The resulting sum, c^s, is quantised with a quantisation parameter QS_Y and with i, j = 0..3 as follows.

再用 QS_Y 对结果 c^s 进行量化，i, j = 0..3，如下所示。

$$c_{ij} = (\ \mathrm{Sign}(\ c_{ij}^s\ ) * (\ \mathrm{Abs}(\ c_{ij}^s\ ) * \mathrm{LevelScale2}(\ QS_Y \,\%\, 6,\ i,\ j\ ) + (\ 1 << (\ 14 + QS_Y / 6\ )\ )\ )\ ) >> (\ 15 + QS_Y / 6\ ) \quad (8\text{-}291)$$

The scaling and transformation process for residual 4x4 blocks as specified in subclause 8.5.8 is invoked with c as the input and x'' as the output.

用 c 作为输入，使用 8.5.8 小节中的 4x4 残差块的缩放和变换过程，然后输出 x''。

The 4x4 array u with elements u_{ij} is derived as follows.

元素为u_{ij}的 4x4 数组 u 的计算如下：

$$u_{ij} = \text{Clip1}(x''_{ij}) \text{ with } i, j = 0..3 \quad (8\text{-}292)$$

The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with luma4x4BlkIdx, u as the input and S' as the output.

将 luma4x4BlkIdx, u 作为输入，使用 8.5.9 小节中的环路滤波之前的图像重建过程，并且输出 S'。

8.6.1.2 Chroma transform coefficient decoding process

8.6.1.2 色差变换系数的解码过程

Inputs to this process are Inter prediction chroma samples for the current macroblock from subclause 8.4 and the prediction residual transform coefficient levels, ChromaDCLevel and ChromaACLevel.

本过程的输入是 8.4 小节中的当前宏块的帧间预测色差象素和预测残差变换系数值 ChromaDCLevel 和 ChromaACLevel。

Outputs of this process are the decoded chroma samples of the current macroblock prior to the deblocking filter process.

本过程的输出环路滤波之前的当前宏块的解码色差象素值。

This process is invoked twice: once for the Cb component and once for the Cr component. The component is referred to by replacing C with Cb for the Cb component and C with Cr for the Cr component. Let iCbCr select the current chroma component.

本过程将被调用两次：Cb 分量一次和 Cr 分量一次。Cb 分量和 Cr 分量都被称为 C 分量，假设 iCbCr 来表示当前的色差分量。

For each 4x4 block of the current chroma component indexed using chroma4x4BlkIdx with chroma4x4BlkIdx equal to 0..3, the following applies.

对于每一个索引为 chroma4x4BlkIdx 的当前色差分量的 4x4 块，chroma4x4BlkIdx 等于 0..3，使用下面的过程。

- The position of the upper-left sample of a 4x4 chroma block with index chroma4x4BlkIdx inside the macroblock is derived as follows

  索引为 chroma4x4BlkIdx 的 4x4 色差块的左上角象素在当前宏块中的位置的计算如下所示：

  $$x = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 0) \quad (8\text{-}293)$$

  $$y = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 1) \quad (8\text{-}294)$$

- Let p be a 4x4 array of prediction samples with element p_{ij} being derived as follows.

  假设 P 为4x4预测象素的数组，元素p_{ij}的计算如下：

  $$p_{ij} = \text{pred}_C[y + j, x + i] \quad (8\text{-}295)$$

- The 4x4 array p is transformed producing transform coefficients c^p(chroma4x4BlkIdx) using Equation 8-286.

  使用等式 8-286 对 4x4 的数组 p 进行变换产生变换系数 c^p(chroma4x4BlkIdx)。

- The variable chromaList, which is a list of 16 entries, is derived. chromaList[0] is set equal to 0. chromaList[0] with index k = 1..15 are specified as follows.

- 变量 chromaList 是一个有 16 个索引的列表，chromaList[0] 被设置成 0，chromaList[0]（k = 1..15）的说明如下：

  $$\text{chromaList}[k] = \text{ChromaACLevel}[\text{iCbCr}][\text{chroma4x4BlkIdx}][k - 1] \quad (8\text{-}296)$$

- The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with chromaList as the input and the 4x4 array c^r as the output.
- 将 chromaList 作为输入，使用 8.5.4 小节中的反向变换系数扫描过程，并且输出 4x4 的数组 c^r。
- The prediction residual transform coefficients c^r are scaled using quantisation parameter QP_C, and added to the transform coefficients of the prediction block c^p with i, j = 0..3 except for the combination i = 0, j = 0 as follows.
- 用量化参数 QP_C 对预测残差变换系数 c^r 进行缩放，并且和预测块的变换系数 c^p 进行相加，其中 i, j = 0..3，除 i = 0, j = 0外，使用下面的过程。

$$c_{ij}^{s} = c_{ij}^{p}(\text{chroma4x4BlkIdx}) + (((c_{ij}^{r} * \text{LevelScale}(QP_C \,\%\, 6, i, j) * A_{ij}) << (QP_C / 6)) >> 6) \qquad (8\text{-}297)$$

- The resulting sum, c^s, is quantised with a quantisation parameter QS_C and with i, j = 0..3 except for the combination i = 0, j = 0 as follows. The derivation of c_{00}(chroma4x4BlkIdx) is described below in this subclause.
- 用量化系数 QS_C 对 c^s 进行量化，其中 i, j = 0..3，除 i = 0, j = 0外，使用下面的过程，c_{00}(chroma4x4BlkIdx) 的计算在本小节的下面进行描述。

$$c_{ij}(\text{chroma4x4BlkIdx}) = (\text{Sign}(c_{ij}^{s}) * (\text{Abs}(c_{ij}^{s}) * \text{LevelScale2}(QS_C \,\%\, 6, i, j) + (1 << (14 + QS_C / 6)))) >> (15 + QS_C / 6) \qquad (8\text{-}298)$$

- The scaling and transformation process for residual 4x4 blocks as specified in subclause 8.5.8 is invoked with c(chroma4x4BlkIdx) as the input and x'' as the output.
- 用 c(chroma4x4BlkIdx) 作为输入，使用 8.5.8 小节中的残差 4x4 块的缩放和变换过程，并且输出 x''。
- The 4x4 array u with elements u_{ij} is derived as follows.
- 元素为 u_{ij} 的 4x4 数组 u 的计算如下：

$$u_{ij} = \text{Clip1}(x''_{ij}) \text{ with } i, j = 0..3 \qquad (8\text{-}299)$$

- The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with chroma4x4BlkIdx and u as the input and S' as the output.
- 用 chroma4x4BlkIdx 和 u 作为输入，使用 8.5.9 小节中的环路滤波之前的图象重建过程，并且输出S'。

The derivation of the DC transform coefficient level c_{00}(chroma4x4BlkIdx) is specified as follows. The DC transform coefficients of the 4 prediction chroma 4x4 blocks of the current component of the macroblock are assembled into a 2x2 matrix of elements c_{00}^{p}(chroma4x4BlkIdx) and a 2x2 transform is applied to the DC transform coefficients as follows

DC 变换系数值 c_{00}(chroma4x4BlkIdx) 的计算说明如下，当前宏块的一个色差分量（Cb 或 Cr）的四个预测色差 4x4 块的 DC 变换系数组合成一个 2x2 的矩阵 c_{00}^{p}(chroma4x4BlkIdx)，并对这个 DC 变换系数进行 2x2 的变换，如下所示：

$$dc^{p} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00}^{p}(0) & c_{00}^{p}(1) \\ c_{00}^{p}(2) & c_{00}^{p}(3) \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad (8\text{-}300)$$

The chroma DC prediction residual transform coefficient levels, ChromaDCLevel[iCbCr][k] with k = 0..3 are scaled using quantisation parameter QP, and added to the prediction DC transform coefficients as follows.

用量化参数 QP 对色差的预测残差 DC 变换系数值 ChromaDCLevel[iCbCr][k] （ k = 0..3）进行缩放操作，并且和预测 DC 变换系数进行相加，如下所示：

$$dc_{ij}^{s} = dc_{ij}^{p} + (((\text{ChromaDCLevel}[\text{iCbCr}][j * 2 + i] * \text{LevelScale}(QP_C \,\%\, 6, 0, 0) * A_{00}) << (QP_C / 6)) >>5) \text{ with } i, j = 0, 1 \qquad (8\text{-}301)$$

The 2x2 array dc^s, is quantised using the quantisation parameter QS_C as follows.

用量化参数 QS_C 对 2x2 数组 dc^s 进行量化操作。

$$dc_{ij}^{r} = (\text{Sign}(dc_{ij}^{s}) * (\text{Abs}(dc_{ij}^{s}) * \text{LevelScale2}(QS_C, 0, 0) + (1 << (15 + QS_C / 6)))) >> (16 + QS_C / 6) \text{ with } i, j = 0, 1 \qquad (8\text{-}302)$$

The 2x2 array f with elements f_{ij} and i, j = 0..1 is derived as follows.

元素为 f_{ij} 的 2x2 数组 f 的计算如下：

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} dc^r_{00} & dc^r_{01} \\ dc^r_{10} & dc^r_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \qquad (8\text{-}303)$$

Scaling of the elements of f is performed as follows.

对 f 中的元素进行缩放操作，如下所示：

- If QS_C is greater than or equal to 6, the $c_{00}(\ )$ are derived as follows.
- 如果QS_C 大于或等于 6，那么 $c_{00}(\ )$ 的计算如下：

$$c_{00}(\ j * 2 + i\ ) = (\ f_{ij} * \text{LevelScale}(\ QS_C \,\%\, 6, 0, 0\ )\ ) << (\ QS_C / 6 - 1\ ) \quad \text{with } i,\ j = 0, 1 \qquad (8\text{-}304)$$

- Otherwise (QS_C is less than 6), the $c_{00}(\ )$ are derived as follows.
- 否则（QS_C 等于 6），那么 $c_{00}(\ )$ 的计算如下：

$$c_{00}(\ j * 2 + i\ ) = (\ f_{ij} * \text{LevelScale}(\ QS_C \,\%\, 6, 0, 0\ )\ ) >> 1 \quad \text{with } i, j = 0, 1 \qquad (8\text{-}305)$$

8.6.2 SP and SI slice decoding process for switching pictures

8.6.2 切换图像的 SP 和 SI slice 的解码过程

This process is invoked, when decoding P macroblock types in SP slices in which sp_for_switch_flag is equal to 1 and when decoding SI macroblock type in SI slices.

在 sp_for_switch_flag 等于 1 的 SP slice 中解码 P 宏块类型或者在 SI slice 中解码 SI 类型宏块时使用这个过程。

Inputs to this process are the prediction residual transform coefficient levels and the prediction sample arrays $pred_L$, $pred_{Cb}$, $pred_{Cr}$ for the current macroblock.

本过程的输入是当前宏块的预测残差变换系数值和预测象素数组 $pred_L$, $pred_{Cb}$, $pred_{Cr}$。

Outputs of this process are the decoded samples of the current macroblock prior to deblocking.

本过程的输出是环路滤波之前的当前宏块的重建象素值。

8.6.2.1 Luma transform coefficient decoding process

8.6.2.1 亮度变换系数的解码过程

Inputs to this process are prediction luma samples $pred_L$ and the luma prediction residual transform coefficient levels, LumaLevel.

本过程的输入是预测亮度象素 $pred_L$ 和亮度预测残差的变换系数值 LumaLevel。

Outputs of this process are the decoded luma samples of the current macroblock prior to the deblocking filter process.

本过程的输出是环路滤波之前当前宏块的解码亮度象素值。

The 4x4 array p with elements p_{ij} with i, j = 0..3 is derived as in subclause 8.6.1.1, is transformed according to Equation 8-286 to produce transform coefficients c^p. These transform coefficients are then quantised with the quantisation parameter QS_Y, as follows:

对 8.6.1.1 小节中的元素为 p_{ij} 的 4x4 数组 P 使用 8-286 等式进行变换产生变换系数 c^p，然后用量化系数 QS_Y 对变换系数进行量化，如下所示。

$$c_{ij}^s = (\ \text{Sign}(\ c_{ij}^p\ ) * (\ \text{Abs}(\ c_{ij}^p\ ) * \text{LevelScale2}(\ QS_Y \,\%\, 6, i, j\ ) + (\ 1 << (\ 14 + QS_Y / 6\ )\ )\ )\ ) >> (\ 15 + QS_Y / 6) \quad \text{with } i, j = 0..3 \qquad (8\text{-}306)$$

The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with LumaLevel[luma4x4BlkIdx] as the input and the two-dimensional array c^r as the output with elements c_{ij}^r.

用 LumaLevel[luma4x4BlkIdx] 作为输入，使用 8.5.4 小节中的反向变换系数扫描过程，输出元素为 c_{ij}^r 的二维数组 c^r。

The 4x4 array c with elements c_{ij} with i, j = 0..3 is derived as follows.

元素为 c_{ij} 的 4x4 数组 c 的计算如下所示。

$$c_{ij} = c_{ij}^{r} + c_{ij}^{s} \quad \text{with } i, j = 0..3 \qquad (8\text{-}307)$$

The scaling and transformation process for residual 4x4 blocks as specified in subclause 8.5.8 is invoked with c as the input and x'' as the output.

将 c 作为输入，使用 8.5.8 小节中的残差 4x4 块的缩放和变换处理，并且输出 x''。

The 4x4 array u with elements u_{ij} is derived as follows.

元素为 u_{ij} 的 4x4 数组 u 的计算如下：

$$u_{ij} = \text{Clip1}(\, x''_{ij}\,) \text{ with } i, j = 0..3 \qquad (8\text{-}308)$$

The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with luma4x4BlkIdx, u as the input and S' as the output.

将 luma4x4BlkIdx, u 作为输入，使用 8.5.9 小节中的环路滤波之前的图像重建过程，并且输出 S'。

8.6.2.2 Chroma transform coefficient decoding process

8.6.2.2 色差变换系数的解码过程

Inputs to this process are predicted chroma samples for the current macroblock from subclause 8.4 and the prediction residual transform coefficient levels, ChromaDCLevel and ChromaACLevel.

本过程的输入是 8.4 小节的当前宏块的预测色差象素和预测残差变换系数值 ChromaDCLevel 和 ChromaACLevel。

Outputs of this process are the decoded chroma samples of the current macroblock prior to the debloc king filter process.

本过程的输出环路滤波之前当前宏块的解码色差值。

This process is invoked twice: once for the Cb component and once for the Cr component. The component is referred to by replacing C with Cb for the Cb component and C with Cr for the Cr component. Let iCbCr select the current chroma component.

本过程被调用两次：Cb 分量一次和 Cr 分量一次。Cb 分量和 Cr 分量都被称为 C 分量，假设 iCbCr 来表示当前的色差分量。

For each 4x4 block of the current chroma component indexed using chroma4x4BlkIdx with chroma4x4BlkIdx equal to 0..3, the following applies.

对于每一个索引为 chroma4x4BlkIdx 的当前色差分量的 4x4 块，chroma4x4BlkIdx 等于 0..3，使用下面的过程。

1. The 4x4 array p with elements p_{ij} with i, j = 0..3 is derived as in subclause 8.6.1.2, is transformed according to Equation 8-286 to produce transform coefficients c^{p}(chroma4x4BlkIdx). These transform coefficients are then quantised with the quantisation parameter QS_C, with i, j = 0..3 except for the combination i = 0, j = 0 as follows. The processing of c_{00}^{p}(chroma4x4BlkIdx) is described below in this subclause.

   对 8.6.1.2 小节中的 元素为 p_{ij} 的4x4 数组 P，i, j = 0..3，使用 8-286 等式进行变换产生变换系数 c^{p}(chroma4x4BlkIdx)，然后再用量化参数 QS_C 对这些变换系数进行量化，i, j = 0..3，除 i = 0, j = 0 之外，如下所示，c_{00}^{p}(chroma4x4BlkIdx) 的处理过程在本小节的下面进行描述。

$$\begin{aligned} c_{ij}^{s} = {} & (\, \text{Sign}(\, c_{ij}^{p}(\, \text{chroma4x4BlkIdx}\,)\,) \; * \\ & (\, \text{Abs}(\, c_{ij}^{p}(\, \text{chroma4x4BlkIdx}\,)\,) * \text{LevelScale2}(\, QS_C \,\%\, 6, i, j\,) + (\, 1 \ll (\, 14 + QS_C / 6\,)\,)\,)\,) \gg (\, 15 + QS_C / 6) \end{aligned} \qquad (8\text{-}309)$$

- The variable chromaList, which is a list of 16 entries, is derived. chromaList[0] is set equal to 0. chromaList[k] with index k = 1..15 are specified as follows.
- 变量 chromaList 是一个有 16 个索引的列表，chromaList[0] 被设置成 0，chromaList[k]（k = 1..15）的说明如下：

$$\text{chromaList}[\, k\,] = \text{ChromaACLevel}[\, \text{iCbCr}\,][\, \text{chroma4x4BlkIdx}\,][\, k - 1\,] \qquad (8\text{-}310)$$

- The inverse transform coefficient scanning process as described in subclause 8.5.4 is invoked with chromaList as the input and the two-dimensional array c^{r}(chroma4x4BlkIdx) as the output with elements c_{ij}^{r}(chroma4x4BlkIdx).

- 将 chromaList 作为输入，使用 8.5.4 小节的反向变换系数扫描处理过程，输出元素为c_{ij}^r(chroma4x4BlkIdx) 的二维数组 c^r(chroma4x4BlkIdx)。
- The 4x4 array c(chroma4x4BlkIdx) with elements c_{ij}(chroma4x4BlkIdx) with i, j = 0..3 except for the combination i = 0, j = 0 is derived as follows. The derivation of c_{00}(chroma4x4BlkIdx) is described below.
- 元素为 c_{ij}(chroma4x4BlkIdx) 的 4x4 数组 c(chroma4x4BlkIdx)的计算如下，i, j = 0..3，i = 0, j = 0 除外，c_{00}(chroma4x4BlkIdx)的计算将在下面描述。

$$c_{ij}(\text{ chroma4x4BlkIdx }) = c_{ij}^r(\text{ chroma4x4BlkIdx }) + c_{ij}^s \qquad (8\text{-}311)$$

- The scaling and transformation process for residual 4x4 blocks as specified in subclause 8.5.8 is invoked with c(chroma4x4BlkIdx) as the input and x’’ as the output.
- 使用 c(chroma4x4BlkIdx) 作为输入，使用 8.5.8 小节中的残差 4x4 块的缩放和变换处理过程，并且输出 x’’。
- The 4x4 array u with elements u_{ij} is derived as follows.
- 元素为 u_{ij} 的 4x4 数组 u 的计算如下：

$$u_{ij} = \text{Clip1}(\ x''_{ij}\ ) \text{ with } i, j = 0..3 \qquad (8\text{-}312)$$

- The picture construction process prior to deblocking filter process in subclause 8.5.9 is invoked with chroma4x4BlkIdx, u as the input and S’ as the output.
- 将 chroma4x4BlkIdx, u 作为输入，使用 8.5.9 小节的环路滤波之前的图像重建处理过程，并且输出 S’。

The derivation of the DC transform coefficient level c_{00}(chroma4x4BlkIdx) is specified as follows. The DC transform coefficients of the 4 prediction 4x4 chroma blocks of the current component of the macroblock, c_{00}^p(chroma4x4BlkIdx), are assembled into a 2x2 matrix of elements and a 2x2 transform is applied to the DC transform coefficients of these blocks according to Equation 8-300 resulting in DC transform coefficients dc_{ij}^p.

DC 变换系数值 c_{00}(chroma4x4BlkIdx) 的计算说明如下，当前宏块的一个色差分量的四个预测 4x4 色差块的DC 系数被组合成一个 2x2 的矩阵，并且根据等式 8-300 对这个矩阵进行 2x2 的变换，产生 DC 变换系数 dc_{ij}^p。

These DC transform coefficients are then quantised with the quantisation parameter QS_C, as given by:

然后用量化参数 QS_C 对这些 DC 变换系数进行量化：

$$dc_{ij}^s = (\text{ Sign}(\ dc_{ij}^p\ ) * (\text{ Abs}(\ dc_{ij}^p\ ) * \text{LevelScale2}(\ QS_C \,\%\, 6, 0, 0\ ) + (\ 1 << (\ 15 + QS_C / 6\ )\ )\ )\ ) >> (\ 16 + QS_C / 6\ ) \quad \text{with } i, j = 0, 1 \qquad (8\text{-}313)$$

The parsed chroma DC prediction residual transform coefficients, ChromaDCLevel[iCbCr][k] with k = 0..3 are added to these quantised DC transform coefficients of the prediction block, as given by:

解析的色差 DC 预测残差变换系数，ChromaDCLevel[iCbCr][k] （ k = 0..3），和这些已经量化的 DC 变换系数进行相加，如下所示：

$$dc_{ij}^r = dc_{ij}^s + \text{ChromaDCLevel}[\text{ iCbCr }][\ j * 2 + i\ ] \quad \text{with } i, j = 0, 1 \qquad (8\text{-}314)$$

The 2x2 array f with elements f_{ij} and i, j = 0..1 is derived using Equation 8-303.

The 2x2 array f with elements f_{ij} and i, j = 0..1 is copied as follows.

元素为 f_{ij} 的 2x2 数组 f 的计算使用等式 8-303，i, j = 0..1。

然后将元素为 f_{ij} 的 2x2 数组 f拷贝到 c_{00}()中。

$$c_{00}(\ j * 2 + i\ ) = f_{ij} \quad \text{with } i, j = 0, 1 \qquad (8\text{-}315)$$

8.7 Deblocking filter process

8.7 环路滤波过程

A conditional filtering shall be applied to all 4x4 block edges of a picture, except edges at the boundary of the picture and any edges for which the deblocking filter process is disabled by disable_deblocking_filter_idc, as specified below. This filtering process shall be performed on a macroblock basis, with all macroblocks in a picture processed in order of increasing macroblock addresses. Prior to the operation of the deblocking filter process for each macroblock, the deblocked samples of the macroblock or macroblock pair above (if any) and the macroblock or macroblock pair to the left (if any) of the current macroblock shall be available.

除了图象的边缘和环路滤波过程由 disable_deblocking_filter_idc 禁止的边缘，图象所有 4x4 块边缘都要进行条件滤波。这个滤波过程以宏块为单位按宏块的地址增加的顺序进行，在对一个宏块环路滤波之前，当前宏块上面宏块或宏块对的已经被滤波的象素和左边宏块或宏块对的已经被滤波的象素都是可用的。

The deblocking filter process is invoked for the luma and chroma components separately. For each macroblock, vertical edges are filtered first, from left to right, and then horizontal edges are filtered from top to bottom. The luma deblocking filter process is performed on four 16-sample edges and the deblocking filter process for each chroma components is performed on two 8-sample edges, for the horizontal direction as shown on the left side of Figure 8-9 and for the vertical direction as shown on the right side of Figure 8-9. Sample values above and to the left of the current macroblock that may have already been modified by the deblocking filter process operation on previous macroblocks shall be used as input to the deblocking filter process on the current macroblock and may be further modified during the filtering of the current macroblock. Sample values modified during filtering of vertical edges are used as input for the filtering of the horizontal edges for the same macroblock.

对色差分量和亮度分量分别使用环路滤波过程，对于一个宏块，先从左到右滤波垂直边缘，然后从上到下滤波水平边缘。对宏块的亮度分量，四条十六个象素的边缘需要进行环路滤波，而色差分量，二条八个象素的边缘需要进行滤波，Figure 8-9的左图表示了水平方向的滤波，右图表示了垂直方向的滤波。在前面宏块的滤波过程中当前宏块的上面或左边的已经被修改的象素值在当前宏块的滤波过程中将作为输入，并且可能被进一步的修改。在垂直边缘滤波的过程中被修改的象素值将作为相同宏块水平滤波的输入象素值，并且可能被进一步的修改。

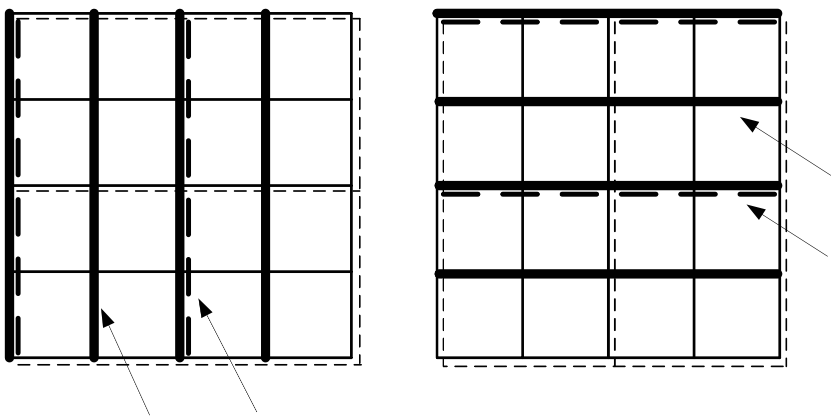


Figure 8-9 – Boundaries in a macroblock to be filtered (luma boundaries shown with solid lines and chroma boundaries shown with dashed lines)

For each macroblock in ascending order of mbAddr, the following applies.

按照 mbAddr 的升序对每一个宏块使用下面的过程。

1. The variables fieldModeMbFlag, filterInternalEdgesFlag, and filterTopMbEdgeFlag are derived as follows.

   变量 fieldModeMbFlag, filterInternalEdgesFlag, 和 filterTopMbEdgeFlag 的计算如下：

   - The variable fieldModeMbFlag is derived as follows.
   - 变量 fieldModeMbFlag 的计算如下：
     - If any of the following conditions is true, fieldModeMbFlag is set equal to 1.

- 如果下面的任何一个条件满足，那么 fieldModeMbFlag 被设置成 1。
  - field_pic_flag is equal to 1
  - field_pic_flag 等于 1
  - MbaffFrameFlag is equal 1 and the macroblock mbAddr is a field macroblock
  - MbaffFrameFlag 等于 1 并且地址为 mbAddr 的宏块是场宏块。
- Otherwise, fieldModeMbFlag is set equal to 0.
- 否则，fieldModeMbFlag 被设置成 0。

- The variable filterInternalEdgesFlag is derived as follows.
- 变量 filterInternalEdgesFlag 的计算如下：
  - If disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is equal to 1, the variable filterInternalEdgesFlag is set equal to 0;
  - 如果包含地址为 mbAddr 的宏块的 slice 的 disable_deblocking_filter_idc 等于 1，那么变量 filterInternalEdgesFlag 将被设置成 0。
  - Otherwise (disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is not equal to 1), the variable filterInternalEdgesFlag is set equal to 1.
  - 否则（包含地址为 mbAddr 的宏块的 slice 的 disable_deblocking_filter_idc 等于 0），变量 filterInternalEdgesFlag 被设置成 0 。
- The variable filterLeftMbEdgeFlag is derived as follows.
- 变量 filterLeftMbEdgeFlag 的计算如下：
  - If any of the following conditions is true, the variable filterLeftMbEdgeFlag is set equal to 0.
  - 如果下面的任何一个条件满足，那么变量 filterLeftMbEdgeFlag 将被设置成 0。
    - the left vertical macroblock edge of the macroblock mbAddr represents a picture boundary
    - 地址为 mbAddr 的宏块的左边垂直边缘是图像的边缘
    - disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is equal to 1
    - 在包含地址为 mbAddr 的宏块的 slice 中 disable_deblocking_filter_idc 等于 1
    - disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is equal to 2 and the left vertical macroblock edge of the macroblock mbAddr represents a slice boundary
    - 在包含地址为 mbAddr 的宏块的 slice 中 disable_deblocking_filter_idc 等于 2 并且地址为 mbAddr 的宏块的左边垂直边缘是 slice 边缘。
  - Otherwise, the variable filterLeftMbEdgeFlag is set equal to 1.
  - 否则，变量 filterLeftMbEdgeFlag 将被设置成 1。
- The variable filterTopMbEdgeFlag is derived as follows.
- 变量 filterTopMbEdgeFlag 的计算如下：
  - If any of the following conditions is true, the variable filterTopMbEdgeFlag is set equal to 0.
  - 如果下面的任何一个条件满足，那么变量 filterTopMbEdgeFlag 将被设置成 0。
    - the top horizontal macroblock edge of the macroblock mbAddr represents a picture boundary
    - 地址为 mbAddr 的宏块的上边水平边缘是图像的边缘
    - disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is equal to 1
    - 在包含地址为 mbAddr 的宏块的 slice 中 disable_deblocking_filter_idc 等于 1
    - disable_deblocking_filter_idc for the slice that contains the macroblock mbAddr is equal to 2 and the top horizontal macroblock edge of the macroblock mbAddr represents a slice boundary

- 在包含地址为 mbAddr 的宏块的 slice 中 disable_deblocking_filter_idc 等于 2 并且地址为 mbAddr 的宏块的上边水平边缘是 slice 边缘。

- Otherwise, the variable filterTopMbEdgeFlag is set equal to 1.
- 否则，变量 filterTopMbEdgeFlag 将被设置成 1。

2. Given the variables fieldModeMbFlag, filterInternalEdgesFlag, and filterTopMbEdgeFlag the deblocking filtering is controlled as follows.

   根据变量 fieldModeMbFlag, filterInternalEdgesFlag, 和 filterTopMbEdgeFlag 的值，按下面的方式对去块斑滤波进行控制。

- When filterLeftMbEdgeFlag is equal to 1, the filtering of the left vertical luma edge is specified as follows.
- 当 filterLeftMbEdgeFlag 等于 1，左边垂直边缘的滤波说明如下：
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (0, k)$ with k = 0..15 as input and S'_L as output.
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (0, k)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
- When filterInternalEdgesFlag is equal to 1, the filtering of the internal vertical luma edges is specified as follows.
- 当 filterInternalEdgesFlag 等于 1，内部垂直亮度边缘的滤波说明如下：
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (4, k)$ with k = 0..15 as input and S'_L as output.
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (8, k)$ with k = 0..15 as input and S'_L as output.
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (12, k)$ with k = 0..15 as input and S'_L as output.
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (4, k)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (8, k)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (12, k)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
- When filterTopMbEdgeFlag is equal to 1, the filtering of the top horizontal luma edge is specified as follows.
- 当 filterTopMbEdgeFlag 等于 1 时，上面水平亮度边缘的滤波说明如下：
  - If MbaffFrameFlag is equal to 1, (mbAddr % 2) is equal to 0, mbAddr is greater than or equal to 2 * PicWidthInMbs, the macroblock mbAddr is a frame macroblock, and the macroblock (mbAddr - 2 * PicWidthInMbs + 1) is a field macroblock, the following applies.
  - 如果 MbaffFrameFlag 等于1, (mbAddr % 2) 等于 0, mbAddr 大于或等于 2 * PicWidthInMbs，宏块 mbAddr 是一帧宏块，而宏块 (mbAddr - 2 * PicWidthInMbs + 1) 为一场宏块，那么使用下面的过程。
    - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, and $(xE_k, yE_k) = (k, 0)$ with k = 0..15 as input and S'_L as output.
    - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, and $(xE_k, yE_k) = (k, 1)$ with k = 0..15 as input and S'_L as output.
    - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, 和 $(xE_k, yE_k) = (k, 0)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
    - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, 和 $(xE_k, yE_k) = (k, 1)$ （k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
  - Otherwise, the process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 0)$ with k = 0..15 as input and S'_L as output.

- 否则，将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (k, 0)$ （ k = 0..15）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。

- When filterInternalEdgesFlag is equal to 1, the filtering of the internal horizontal luma edges is specified as follows.
- 当 filterInternalEdgesFlag 等于 1 时，内部水平亮度边缘的滤波说明如下：
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 4)$ with k = 0..15 as input and S'_L as output.
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 8)$ with k = 0..15 as input and S'_L as output.
  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 12)$ with k = 0..15 as input and S'_L as output.
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (k, 4)$ （ k = 0..15 ）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (k, 8)$ （ k = 0..15 ）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
  - 将 mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (k, 12)$ （ k = 0..15 ）作为输入，使用 8.7.1 小节的过程，并且输出 S'_L 。
- For both chroma components iCbCr = 0 and 1, the following applies.
- 对于两个色差分量 iCbCr = 0 和 1，使用下面的过程。
  - When filterLeftMbEdgeFlag is equal to 1, the filtering of the left vertical chroma edge is specified as follows.
  - 当 filterLeftMbEdgeFlag 等于 1，左边垂直色差边缘的滤波说明如下：
    - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (0, k)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.
    - 将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (0, k)$ （ k = 0..7）作为输入，使用使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。
  - When filterInternalEdgesFlag is equal to 1, the filtering of the internal vertical chroma edge is conducted as follows
  - 当 filterInternalEdgesFlag 等于 1 时，内部垂直色差边缘的滤波说明如下：
    - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (4, k)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.
    - 将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag, 和 $(xE_k, yE_k) = (4, k)$ （ k = 0..7）作为输入，使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。
  - When filterTopMbEdgeFlag is equal to 1, the filtering of the top horizontal chroma edge is specified as follows.
  - 当 filterTopMbEdgeFlag 等于 1 时，上面水平色差边缘的滤波说明如下：
    - If MbaffFrameFlag is equal to 1, (mbAddr % 2) is equal to 0, mbAddr is greater than or equal to 2 * PicWidthInMbs, the macroblock mbAddr is a frame macroblock, and the macroblock (mbAddr – 2 * PicWidthInMbs + 1) is a field macroblock, the following applies.
    - 如果 MbaffFrameFlag 等于1， (mbAddr % 2) 等于0， mbAddr 大于或等于 2 * PicWidthInMbs, 宏块 mbAddr 是帧宏块，而宏块 (mbAddr – 2 * PicWidthInMbs + 1) 是场宏块，那么使用下面的过程。

- The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, and $(xE_k, yE_k) = (k, 0)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.

- The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, and $(xE_k, yE_k) = (k, 1)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.

- 将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, 和 $(xE_k, yE_k) = (k, 0)$ （k = 0..7 ）作为输入，使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。

- 将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1, 和 $(xE_k, yE_k) = (k, 1)$ （k = 0..7 ）作为输入，使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。

- Otherwise, the process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 0)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.

- 否则，将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 0)$ （ k = 0..7）作为输入，使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。

- When filterInternalEdgesFlag is equal to 1, the filtering of the internal horizontal chroma edge is specified as follows.

- 当 filterInternalEdgesFlag 等于 1 时，中间水平色差边缘的滤波说明如下：

  - The process specified in subclause 8.7.1 is invoked with mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 4)$ with k = 0..7 as input and S'_C with C being replaced by Cb for iCbCr = 0 and C being replaced by Cr for iCbCr = 1 as output.

  - 将 mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag, and $(xE_k, yE_k) = (k, 4)$ （ k = 0..7）作为输入，使用 8.7.1 小节的过程，然后输出 S'_C，当 iCbCr = 0 时，c 由 Cb 取代，当 iCbCr = 1 时，c 由 Cr 取代。

NOTE - When field mode filtering (fieldModeFilteringFlag is equal to 1) is applied across the top horizontal edges of a frame macroblock, this vertical filtering across the top or bottom macroblock boundary may involve some samples that extend across an internal block edge that is also filtered internally in frame mode.

注 一 当在一帧宏块的上面水平边缘进行场模式的滤波时（fieldModeFilteringFlag 等于 1），上面或者下面宏块边缘的垂直滤波可能涉及到在帧模式滤波中已被滤波的内部块边缘的象素。

NOTE – In all cases, 3 horizontal luma edges, 1 horizontal chroma edge for Cb, and 1 horizontal chroma edge for Cr are filtered that are internal to a macroblock. When field mode filtering (fieldModeFilteringFlag is equal to 1) is applied to the top edges of a frame macroblock, 2 horizontal luma, 2 horizontal chroma edges for Cb, and 2 horizontal chroma edges for Cr between the frame macroblock and the above macroblock pair are filtered using field mode filtering, for a total of up to 5 horizontal luma edges, 3 horizontal chroma edges for Cb, and 3 horizontal chroma edges for Cr filtered that are considered to be controlled by the frame macroblock. In all other cases, at most 4 horizontal luma, 2 horizontal chroma edges for Cb, and 2 horizontal chroma edges for Cr are filtered that are considered to be controlled by a particular macroblock.

注 一 在宏块的内部，需要对 3 个水平亮度边缘，一个水平色差 Cb 边缘和一个水平色差 Cr 边缘进行滤波。当对帧宏块的上边缘进行场模式滤波时（fieldModeFilteringFlag 等于 1），在帧宏块和上面的宏块对之间有2个水平亮度边缘，2个水平色差 Cb 边缘 和 2 个水平色差 Cr 边缘需要用场模式进行滤波，因此对于帧宏块，总共有 5 个水平亮度边缘、3 个水平色差 Cb 边缘和 3 个水平色差 Cr 边缘需要滤波。对于一个具体的宏块，最多有 4 个水平亮度边缘，2 个水平色差 Cb 边缘和 2 个水平色差 Cr 边缘需要滤波！

Finally, the arrays S'_L, S'_{Cb}, S'_{Cr} are assigned to the arrays S_L, S_{Cb}, S_{Cr} (which represent the decoded picture), respectively.

最后，数组S'_L, S'_{Cb}, S'_{Cr}分别赋给 S_L, S_{Cb}, S_{Cr}（表示解码图像）。

8.7.1 Filtering process for block edges

8.7.1 块边缘的滤波过程

Input to this process are mbAddr, chromaEdgeFlag, the chroma component index iCbCr (if chromaEdgeFlag is equal to 1), verticalEdgeFlag, fieldModeFilteringFlag, and a set of sixteen luma (if chromaEdgeFlag is equal to 0) or eight chroma (if chromaEdgeFlag is equal to 1) sample locations (xE_k, yE_k), with k = 0 .. nE - 1, expressed relative to the upper left corner of the macroblock mbAddr. The set of sample locations (xE_k, yE_k) represent the sample locations immediately to the right of a vertical edge (if verticalEdgeFlag is equal to 1) or immediately below a horizontal edge (if verticalEdgeFlag is equal to 0).

本过程的输入是：mbAddr, chromaEdgeFlag，色差分量索引 iCbCr（如果 chromaEdgeFlag 等于 1），verticalEdgeFlag, fieldModeFilteringFlag，和 16 个亮度系数坐标位置（如果 chromaEdgeFlag 等于 0）或者 8 个色差系数坐标位置（如果 chromaEdgeFlag 等于 1）(xE_k, yE_k) ($k = 0 .. nE – 1$)，这些坐标位置都是相对与宏块 mbAddr 左上角象素的坐标位置，它们表示了紧邻垂直边缘右边象素的坐标位置（如果 verticalEdgeFlag 等于 1）或者紧邻水平边缘下边象素的坐标位置（如果 verticalEdgeFlag 等于 0）。

The variable nE is derived as follows.

变量 nE 的计算如下：

- If chromaEdgeFlag is equal to 0, nE is 16;
- Otherwise (chromaEdgeFlag is equal to 1), nE is 8.
- 如果 chromaEdgeFlag 等于 0，那么 nE 等于 16；
- 否则（chromaEdgeFlag 等于 1），nE 等于 8；

Let s' be a variable specifying a luma or chroma sample array, be derived as follows.

假设 s' 是一个表示亮度或者色差象素数组的变量，计算如下：

- If chromaEdgeFlag is equal to 0, s' represents the luma sample array S'_L of the current picture.
- If chromaEdgeFlag is equal to 1 and iCbCr is equal to 0, s' represents the chroma sample array S'_{Cb} of the chroma component Cb of the current picture.
- Otherwise (chromaEdgeFlag is equal to 1 and iCbCr is equal to 1), s’ represents the chroma sample array S'_{Cr} of the chroma component Cr of the current picture.
- 如果 chromaEdgeFlag 等于 0，那么 s' 表示当前图像中的亮度象素数组 S'_L。
- 如果 chromaEdgeFlag 等于 1 并且 iCbCr 等于 0，那么 s' 表示当前图像中色差 Cb 分量的色差象素数组 S'_{Cb}。
- 否则（chromaEdgeFlag 等于 1 并且 iCbCr 等于 1），那么 s' 表示当前图像中色差 Cr 分量的色差象素数组 S'_{Cr}。

The variable dy is derived as follows.

变量 dy 的计算如下：

- If fieldModeFilteringFlag is equal to 1 and MbaffFrameFlag is equal to 1, dy is set equal to 2.
- Otherwise (fieldModeFilteringFlag is equal to 0 or MbaffFrameFlag is equal to 0), dy is set equal to 1.
- 如果 fieldModeFilteringFlag 等于 1 并且 MbaffFrameFlag 等于 1, 那么 dy 将被设置成 2。
- 否则（fieldModeFilteringFlag 等于 0 或者 MbaffFrameFlag 等于 0），那么dy 将被设置成 1。

The position of the upper-left luma sample of the macroblock mbAddr is derived by invoking the inverse macroblock scanning process in subclause 6.4.1 with mbAddr as input and the output being assigned to (xP, yP).

将 mbAddr 作为输入，使用 6.4.1 小节中的反向宏块扫描过程，计算出宏块 mbAddr 中左上角象素的坐标位置并且输出 (xP, yP)。

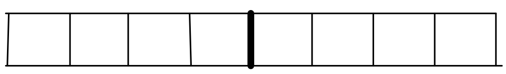


Figure 8-10 – Convention for describing samples across a 4x4 block horizontal or vertical boundary

For each sample location (xE_k, yE_k), $k = 0 .. nE - 1$, the following applies.

对于每一个象素位置 (xE_k, yE_k)，$k = 0 .. nE – 1$，使用下面的过程。

- The filtering process is applied to a set of eight samples across a 4x4 block horizontal or vertical edge denoted as p_i and q_i with i = 0..3 as shown in Figure 8-10 with the edge lying between p_0 and q_0. p_i and q_i with i = 0..3 are specified as follows.
- 对横跨 4x4 块水平边缘或者垂直边缘的八个象素进行滤波，如图 Figure 8-10 中的 p_i 和 q_i，i = 0..3，块的边缘位于 p_0 和 q_0 之间，p_i 和 q_i 的说明如下：
  - If verticalEdgeFlag is equal to 1,

    如果 verticalEdgeFlag 等于 1

    $$q_i = s'[\ xP + xE_k + i,\ yP + yE_k\ ] \quad (8\text{-}316)$$

    $$p_i = s'[\ xP + xE_k - i - 1,\ yP + yE_k\ ] \quad (8\text{-}317)$$

  - Otherwise (verticalEdgeFlag is equal to 0),

    否则（ verticalEdgeFlag 等于 0）

    $$q_i = s'[\ xP + xE_k,\ yP + dy * (\ yE_k + i\ )\ ] \quad (8\text{-}318)$$

    $$p_i = s'[\ xP + xE_k,\ yP + dy * (\ yE_k - i - 1\ )\ ] \quad (8\text{-}319)$$

- The process specified in subclause 8.7.2 is invoked with the sample values p_i and q_i (i = 0..3), chromaEdgeFlag, verticalEdgeFlag, and fieldModeFilteringFlag as input, and the output is assigned to the filtered results sample values p'_i and q'_i with i = 0..2.

  将象素值 p_i 和 q_i (i = 0..3)、chromaEdgeFlag、verticalEdgeFlag 和 fieldModeFilteringFlag 作为输入，使用 8.7.2 小节中的处理过程，输出滤波后的象素值 p'_i 和 q'_i （ i = 0..2）。

- The input sample values p_i and q_i with i = 0..2 are replaced by the corresponding filtered result sample values p'_i and q'_i with i = 0..2 inside the sample array s' as specified in the following.

  在象素数组 s' 中输入的象素值 p_i 和 q_i （ i = 0..2）将被对应的滤波后的象素值 p'_i 和 q'_i （ i = 0..2） 替代。

  - If verticalEdgeFlag is equal to 1,

    如果 verticalEdgeFlag 等于 1，

    $$s'[\ xP + xE_k + i,\ yP + yE_k\ ] = q'_i \quad (8\text{-}320)$$

    $$s'[\ xP + xE_k - i - 1,\ yP + yE_k\ ] = p'_i \quad (8\text{-}321)$$

  - Otherwise (verticalEdgeFlag is equal to 0),

    否则（ verticalEdgeFlag 等于 0），

    $$s'[\ xP + xE_k,\ yP + dy * (\ yE_k + i\ )\ ] = q'_i \quad (8\text{-}322)$$

    $$s'[\ xP + xE_k,\ yP + dy * (\ yE_k - i - 1\ )\ ] = p'_i \quad (8\text{-}323)$$

8.7.2 Filtering process for a set of samples across a horizontal or vertical block edge

8.7.2 水平块边缘或垂直块边缘象素的滤波过程

Inputs to this process are the input sample values p_i and q_i with i in the range of 0..3 of a single set of samples across an edge that is to be filtered, chromaEdgeFlag, verticalEdgeFlag, and fieldModeFilteringFlag.

本过程的输入是横跨边缘的象素 p_i and q_i （ i ＝ 0..3 ）、chromaEdgeFlag、verticalEdgeFlag 和 fieldModeFilteringFlag。

Outputs of this process are the filtered result sample values p'_i and q'_i with i in the range of 0..2.

本过程的输出是滤波后的象素值 p'_i 和 q'_i （ i ＝ 0..2）。

The content dependent boundary filtering strength variable bS is derived as follows.

和内容相关的边缘滤波权重 bS 的计算如下：

- If chromaEdgeFlag is equal to 0, the derivation process for the content dependent boundary filtering strength specified in subclause 8.7.2.1 is invoked with p_0, q_0, and verticalEdgeFlag as input, and the output is assigned to bS.

  如果 chromaEdgeFlag 等于 0，那么将 p_0，q_0 和 verticalEdgeFlag 作为输入，使用 8.7.2.1 小节中的计算和内容相关的边缘滤波权重 bS的过程，输出分配给 bS。

- Otherwise (chromaEdgeFlag is equal to 1), the following applies.

  否则（chromaEdgeFlag 等于 1），使用下面的过程

  - If fieldModeFilteringFlag is equal to 0, the bS used for filtering a set of samples of a horizontal or vertical chroma edge shall be set equal to the value of bS for filtering the set of samples of a horizontal or vertical luma edge, respectively, that contains the luma sample at location (2 * x, 2 * y) inside the luma array of the frame, where (x, y) is the location of the chroma sample q_0 inside the chroma array for that frame.
  - 如果 fieldModeFilteringFlag 等于 0，那么滤波水平或垂直色差边缘的 bS 值将分别被设置成滤波水平或垂直亮度边缘的 bS 值，这里的亮度边缘在帧中的位置为 (2 * x, 2 * y)，(x, y) 表示在帧色差数组中色差象素 q_0 的位置。
  - Otherwise (fieldModeFilteringFlag is equal to 1), the bS used for filtering a set of samples of a horizontal or vertical chroma edge shall be set equal to the value of bS for filtering the set of samples of a horizontal or vertical luma edge, respectively, that contains the luma sample at location (2 * x, 2 * y) inside the luma array of the same field, where (x, y) is the location of the chroma sample q_0 inside the chroma array for that field.
  - 否则（fieldModeFilteringFlag 等于 1），那么滤波水平或垂直色差边缘的 bS 值将分别被设置成滤波水平或垂直亮度边缘的 bS 值，这里的亮度边缘在对应场中的位置为 (2 * x, 2 * y)，(x, y) 表示在场色差数组中色差象素 q_0 的位置。

The process specified in subclause 8.7.2.2 is invoked with p_0, q_0, p_1, q_1, chromaEdgeFlag, and bS as input, and the output is assigned to filterSamplesFlag, indexA, α, and β.

将 p_0、q_0、p_1、q_1、chromaEdgeFlag 和 bS 作为输入，使用 8.7.2.2 小节中的处理过程，输出赋给 filterSamplesFlag、indexA、α 和 β

Depending on the variable filterSamplesFlag, the following applies.

根据变量 filterSamplesFlag 的值，分别使用下面的过程。

- If filterSamplesFlag is equal to 1, the following applies.
- 如果 filterSamplesFlag 等于 1，使用下面的过程
  - If bS is less than 4, the process specified in subclause 8.7.2.3 is invoked with p_i and q_i (i = 0..3), chromaEdgeFlag, bS, β, and indexA given as input, and the output is assigned to p'_i and q'_i (i = 0..2).

    如果 bS 小于 4，那么将 p_i 和 q_i (i = 0..3), chromaEdgeFlag, bS, β, 和 indexA 作为输入，使用 8.7.2.3 小节中的过程，输出赋给 p'_i 和 q'_i (i = 0..2)。
  - Otherwise (bS is equal to 4), the process specified in subclause 8.7.2.4 is invoked with p_i and q_i (i = 0..3), chromaEdgeFlag, α, and β given as input, and the output is assigned to p'_i and q'_i (i = 0..2).

    否则（bS 等于 4），那么将 p_i 和 q_i (i = 0..3), chromaEdgeFlag, α, 和 β 作为输入，使用 8.7.2.4 小节中的过程，输出赋给 p'_i 和 q'_i (i = 0..2)。
- Otherwise (filterSamplesFlag is equal to 0), the filtered result samples p'_i and q'_i (i = 0..2) are replaced by the corresponding input samples p_i and q_i:
- 否则（filterSamplesFlag 等于 0），滤波后的象素p'_i 和 q'_i (i = 0..2) 被对应的输入象素 p_i 和 q_i 取代。

for i = 0..2, $p'_i = p_i$ (8-324)

for i = 0..2, $q'_i = q_i$ (8-325)

8.7.2.1 Derivation process for the luma content dependent boundary filtering strength

8.7.2.1 对与内容相关的边缘滤波权重的处理过程

Inputs to this process are the input sample values p_0 and q_0 of a single set of samples across an edge that is to be filtered and verticalEdgeFlag.

本过程的输入是横跨被滤波边缘的象素 p_0 和 q_0，和 verticalEdgeFlag。

Output of this process is the variable bS.

本过程的输出是变量 bS 。

Let the variable mixedModeEdgeFlag be derived as follows.

假设变量 mixedModeEdgeFlag 计算如下：

- If MbaffFrameFlag is equal to 1 and the samples p_0 and q_0 are in different macroblock pairs, one of which is a field macroblock pair and the other is a frame macroblock pair, mixedModeEdgeFlag is set equal to 1
- Otherwise, mixedModeEdgeFlag is set equal to 0.
- 如果 MbaffFrameFlag 等于 1，并且象素 p_0 和 q_0 在不同的宏块对中，其中的一个宏块对为场宏块对，另一宏块对为帧宏块对，那么 mixedModeEdgeFlag 将被设置成 1；
- 否则，mixedModeEdgeFlag 将被设置成 0。

If the block edge is also a macroblock edge and any of the following conditions are true, a value of bS equal to 4 shall be the output:

如果块边缘也是宏块的边缘，并且下面的任何一个条件满足，那么 bS 的值将被设置成 4 并输出。

– the samples p_0 and q_0 are both in frame macroblocks and either of the macroblocks containing samples p_0 and q_0 is coded using an Intra macroblock prediction mode
– MbaffFrameFlag is equal to 1, verticalEdgeFlag is equal to 1, and either of the macroblocks containing samples p_0 and q_0 is coded using an Intra macroblock prediction mode.
– 象素 p_0 和 q_0 都在帧宏块中，并且包含象素 p_0 和 q_0 的宏块中至少一个宏块是帧内宏块预测模式；
– MbaffFrameFlag 等于 1，verticalEdgeFlag 等于 1，并且包含象素 p_0 和 q_0 的宏块中至少一个宏块是帧内宏块预测模式；

If any of the following conditions are true, a value of bS equal to 3 shall be the output:

如果下面的任何一个条件满足，那么 bS 的值将被设置成 3 并且输出：

– mixedModeEdgeFlag is equal to 0 and either of the macroblocks containing samples p_0 and q_0 is coded using an Intra macroblock prediction mode
– mixedModeEdgeFlag is equal to 1, verticalEdgeFlag is equal to 0, and either of the macroblocks containing samples p_0 and q_0 is coded using an Intra macroblock prediction mode
– mixedModeEdgeFlag 等于 0 并且包含象素 p_0 和 q_0 的宏块中至少一个宏块是帧内宏块预测模式；
– mixedModeEdgeFlag 等于1 ，verticalEdgeFlag 等于 0，并且包含象素 p_0 和 q_0 的宏块中至少一个宏块是帧内宏块预测模式；

If the following condition is true, a value of bS equal to 2 shall be the output:

如果下面的任何一个条件满足，那么 bS 的值将被设置成 2 并且输出：

– the 4x4 luma block containing sample p_0 or the 4x4 luma block containing sample q_0 contains non-zero transform coefficient levels
– 包含象素 p_0 的 4x4 亮度块或者包含象素 p_1 的 4x4 亮度块中含有非零变换系数值。

If any of the following conditions are true, a value of bS equal to 1 shall be the output:

如果下面的任何一个条件满足，那么 bS 的值将被设置成 1 并且输出：

– mixedModeEdgeFlag is equal to 1
– mixedModeEdgeFlag 等于1
– mixedModeEdgeFlag is equal to 0 and for the prediction of the macroblock partition containing the sample p_0 different reference pictures or a different number of reference pictures are used than for the prediction of the macroblock partition containing the sample q_0.
– MixedModeEdgeFlag 等于 0，在预测包含象素 p_0 的宏块部分时和预测包含象素 q_0 的宏块部分时使用了不同的参考图像。
– mixedModeEdgeFlag is equal to 0 and one motion vector is used to predict the macroblock/sub-macroblock partition containing the sample p_0 and one motion vector is used to predict the macroblock/sub-macroblock partition

containing the sample q_0 and the absolute difference between the horizontal or vertical component of the motion vector used is greater than or equal to 4 in units of quarter luma frame samples.

– MixedModeEdgeFlag 等于 0，用来预测包含象素 p_0 的宏块部分运动矢量的运动矢量和用来预测包含象素 p_1 的宏块部分运动矢量的运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个 1/4亮度象素。

– mixedModeEdgeFlag is equal to 0 and two motion vectors and two different reference pictures are used to predict the macroblock/sub-macroblock partition containing the sample p_0 and two motion vectors for the same two reference pictures are used to predict the macroblock/sub-macroblock partition containing the sample q_0 and the absolute difference between the horizontal or vertical component of a motion vector used in the prediction of the two the macroblock/sub-macroblock partitions for the same reference picture is greater than or equal to 4 in units of quarter luma frame samples.

– MixedModeEdgeFlag 等于 0，两个运动矢量和两个不同的参考图像被用来预测包含象素 p_0 的宏块/子宏块部分，两个运动矢量和两个相同的参考图像被用来预测包含象素 p_1 的宏块/子宏块部分，用相同的参考图像来预测两个宏块/子宏块部分时使用的两个运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个 1/4 亮度象素。

– mixedModeEdgeFlag is equal to 0 and two motion vectors for the same reference picture are used to predict the macroblock/sub-macroblock partition containing the sample p_0 and two motion vectors for the same reference picture as used to predict the macroblock/sub-macroblock partition containing the sample p_0 are used to predict the macroblock/sub-macroblock partition containing the sample q_0 and both of the following conditions are true:

– MixedModeEdgeFlag 等于 0，对于相同的参考帧，用两个运动矢量用来预测包含p_0 的宏块/子宏块部分，对于同样的参考图像，也用两个运动矢量来预测包含p_1 的宏块/子宏块部分，并且下面的两个条件都满足：

  – The absolute difference between the horizontal or vertical component of list 0 motion vectors used in the prediction of the two macroblock/sub-macroblock partitions is greater than or equal to 4 in quarter luma frame samples or the absolute difference between the horizontal or vertical component of the list 1 motion vectors used in the prediction of the two macroblock/sub-macroblock partitions is greater than or equal to 4 in units of quarter luma frame samples.

    在预测两个宏块/子宏块部分时列表 0 的运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个 1/4 亮度象素或者在预测两个宏块/子宏块部分时列表 1 的运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个1/4 亮度象素。

  – The absolute difference between the horizontal or vertical component of list 0 motion vector used in the prediction of the macroblock/sub-macroblock partition containing the sample p_0 and the list 1 motion vector used in the prediction of the macroblock/sub-macroblock partition containing the sample q_0 is greater than or equal to 4 in units of quarter luma frame samples or the absolute difference between the horizontal or vertical component of the list 1 motion vector used in the prediction of the macroblock/sub-macroblock partition containing the sample p_0 and list 0 motion vector used in the prediction of the macroblock/sub-macroblock partition containing the sample q_0 is greater than or equal to 4 in units of quarter luma frame samples.

    预测包含 p_0 的宏块/子宏块的列表 0 的运动矢量的水平分量或者垂直分量和预测包含 p_1 的宏块/子宏块的列表 1 的运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个1/4 亮度象素，或者预测包含 p_0 的宏块/子宏块的列表 1 的运动矢量的水平分量或者垂直分量和预测包含 p_1 的宏块/子宏块的列表 0 的运动矢量的水平分量或者垂直分量的绝对差大于或者等于 4 个1/4 亮度象素，

NOTE – A vertical difference of 4 in units of quarter luma frame samples is a difference of 2 in units of quarter luma field samples

注 － 4 个1/4 亮度帧象素的垂直差和2 个1/4 亮度场象素的垂直差是不同的。

Otherwise, a value of bS equal to 0 shall be the output.

否则，bS 的值将被设置成 0 并且输出。

8.7.2.2 Derivation process for the thresholds for each block edge

8.7.2.2 对于每个块边缘阈值的计算过程

Inputs to this process are the input sample values p_0, q_0, p_1 and q_1 of a single set of samples across an edge that is to be filtered, chromaEdgeFlag, and bS, for the set of input samples, as specified in 8.7.2.

本过程的输入是横跨被滤波边缘的象素 p_0, q_0, p_1 和 q_1，以及bS。

Outputs of this process are the variable filterSamplesFlag, which indicates whether the input samples are filtered, the value of indexA, and the values of the threshold variables α and β.

本过程的输出是变量 filterSamplesFlag，表示输入象素是否需要滤波，indexA 的值和阈值变量 α 和 β 的值。

Let qP_p and qP_q be variables specifying quantisation parameter values for the macroblocks containing the samples p_0 and q_0, respectively. The variables qP_z (with z being replaced by p or q) are derived as follows.

假设 qP_p 和 qP_q 是包含 p_0 和 q_0 的宏块的量化参数值，那么变量 qP_z 的计算如下：

- If chromaEdgeFlag is equal to 0, the following applies.
- 如果 chromaEdgeFlag 等于 0，使用下面的过程：
  - If the macroblock containing the sample z_0 is an I_PCM macroblock, qP_z is set to 0.
  - Otherwise (the macroblock containing the sample z_0 is not an I_PCM macroblock), qP_z is set to the value of QP_Y of the macroblock containing the sample z_0.
  - 如果包含象素 z_0 的宏块是 I_PCM 宏块，那么 qP_z 被设置成 0。
  - 否则（包含象素 z_0 的宏块不是 I_PCM 宏块），qP_z 被设置成象素 z_0 的宏块的 QP_Y 值。
- Otherwise (chromaEdgeFlag is equal to 1), the following applies.
- 否则（ chromaEdgeFlag 等于 1），使用下面的过程
  - If the macroblock containing the sample z_0 is an I_PCM macroblock, qP_z is set to the value of QP_C that corresponds to a value of 0 for QP_Y as specified in subclause 8.5.5.
  - 如果包含象素 z_0 的宏块是一个 I_PCM 宏块，那么 qP_z 被设置成对应于宏块 QP_Y ＝ 0的QP_C值，如 8.5.5 所示。
  - Otherwise (the macroblock containing the sample z_0 is not an I_PCM macroblock), qP_z is set to the value of QP_C that corresponds to the value QP_Y of the macroblock containing the sample z_0 as specified in subclause 8.5.5.
  - 否则（包含象素 z_0 的宏块不是一个 I_PCM 宏块），qP_z 被设置对应于成包含象素 z_0 的宏块 QP_Y 值的QP_C 值，如 8.5.5 所示。

Let qPav be a variable specifying an average quantisation parameter. It is derived as follows.

假设 qPav 是一个表示平均量化参数的一个变量，它的计算如下：

$$qPav = (qP_p + qP_q + 1) >> 1 \quad (8\text{-}326)$$

NOTE - In SP and SI slices, qP_{av} is derived in the same way as in other slice types. QS_Y from Equation 7-17 is not used in the deblocking filter.
注 － 在 SP 和 SI slices 中，qP_{av} 按其他 slice 类型中的方法进行计算。等式 7-17 中的 QS_Y 在环路滤波中没有使用。

Let indexA be a variable that is used to access the α table (Table 8-14) as well as the t_{C0} table (Table 8-15), which is used in filtering of edges with bS less than 4 as specified in subclause 8.7.2.3, and let indexB be a variable that is used to access the β table (Table 8-14). The variables indexA and indexB are derived as follows, where the values of FilterOffsetA and FilterOffsetB are the values of those variables specified in subclause 7.4.3 for the slice that contains the macroblock containing sample q_0..

假设 indexA 是一个访问 α 表（Table 8-14）和 t_{C0} 表（Table 8-15）的变量，这些表在滤波 bS 值小于 4 的边缘时使用，如 8.7.2.3 所示，indexB 是一个访问 β 表的变量（Table 8-14），变量 indexA 和 indexB 的计算如下，这里 FilterOffsetA 和 FilterOffsetB 是 7.4.3 小节中对于包含包含象素 q_0 的宏块的 slice 的说明变量，

$$indexA = Clip3(0, 51, qP_{av} + FilterOffsetA) \quad (8\text{-}327)$$

$$indexB = Clip3(0, 51, qP_{av} + FilterOffsetB) \quad (8\text{-}328)$$

The threshold variables α and β are specified in Table 8-14 depending on the values of indexA and indexB.

根据变量 indexA 和 indexB 的值，阈值变量 α 和 β 的说明如表 Table 8-14 所示。

The variable filterSamplesFlag is derived by

变量 filterSamplesFlag 的计算如下：

$$filterSamplesFlag = (bS \mathrel{!=} 0 \;\&\&\; Abs(p_0 - q_0) < \alpha \;\&\&\; Abs(p_1 - p_0) < \beta \;\&\&\; Abs(q_1 - q_0) < \beta) \quad (8\text{-}329)$$

Table 8-14 – Derivation of indexA and indexB from offset dependent threshold variables α and β

	indexA (for α) or indexB (for β)																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
α	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	6	7	8	9	10	12	13
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	3	3	3	3	4	4	4

Table 8-14 (concluded) – Derivation of indexA and indexB from offset dependent threshold variables α and β

	indexA (for α) or indexB (for β)																									
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
α	15	17	20	22	25	28	32	36	40	45	50	56	63	71	80	90	101	113	127	144	162	182	203	226	255	255
β	6	6	7	7	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18

8.7.2.3 Filtering process for edges with bS less than 4

8.7.2.3 对 bS 小于 4 的边缘的滤波过程

Inputs to this process are the input sample values p_i and q_i ($i = 0..3$) of a single set of samples across an edge that is to be filtered, chromaEdgeFlag, bS, β, and indexA, for the set of input samples, as specified in 8.7.2.

本过程的输入是横跨被滤波边缘的象素 p_i 和 q_i ($i = 0..3$)、chromaEdgeFlag、bS、β 和 indexA。

Outputs of this process are the filtered result sample values p'_i and q'_i ($i = 0..2$) for the set of input sample values.

本过程的输出是滤波后的象素值 p'_i 和 q'_i ($i = 0..2$)。

The filtered result samples p'_0 and q'_0 are derived by

滤波后的象素值 p'_0 and q'_0 的计算如下：

$$\Delta = \mathrm{Clip3}(-t_C, t_C, ((((q_0 - p_0) << 2) + (p_1 - q_1) + 4) >> 3)) \quad (8\text{-}330)$$

$$p'_0 = \mathrm{Clip1}(p_0 + \Delta) \quad (8\text{-}331)$$

$$q'_0 = \mathrm{Clip1}(q_0 - \Delta) \quad (8\text{-}332)$$

where the threshold t_C is determined as follows.

这里的阈值 t_C 的计算如下：

- If chromaEdgeFlag is equal to 0,
- 如果 chromaEdgeFlag 等于 0，

$$t_C = t_{C0} + ((a_p < \beta) ? 1 : 0) + ((a_q < \beta) ? 1 : 0) \quad (8\text{-}333)$$

- Otherwise (chromaEdgeFlag is equal to 1),
- 否则（chromaEdgeFlag 等于 1），

$$t_C = t_{C0} + 1 \quad (8\text{-}334)$$

The threshold t_{C0} is specified in Table 8-15 depending on the values of indexA and bS.

根据 indexA 和 bS 的值，表 Table 8-15 中的阈值 t_{C0} 的说明如下：

Let a_p and a_q be two threshold variables specified by

假设 a_p 和 a_q 是两个阈值变量，说明如下：

$$a_p = \mathrm{Abs}(p_2 - p_0) \quad (8\text{-}335)$$

$$a_q = \mathrm{Abs}(q_2 - q_0) \quad (8\text{-}336)$$

The filtered result sample p'_1 is derived as follows

滤波后的象素 p'_1 的计算如下：

- If chromaEdgeFlag is equal to 0 and a_p is less than β,
- 如果 chromaEdgeFlag 等于 0 并且 a_p 小于 β，

$$p'_1 = p_1 + \mathrm{Clip3}(-t_{C0}, t_{C0}, (p_2 + ((p_0 + q_0 + 1) >> 1) - (p_1 << 1)) >> 1) \quad (8\text{-}337)$$

- Otherwise (chromaEdgeFlag is equal to 1 or a_p is greater than or equal to β),
- 否则（chromaEdgeFlag 等于 1 或者 a_p 大于等于 β，）

$$p'_1 = p_1 \quad (8\text{-}338)$$

The filtered result sample q'_1 is derived as follows

滤波后的象素 q'_1 的计算如下：

- If chromaEdgeFlag is equal to 0 and a_q is less than β,
- 如果 chromaEdgeFlag 等于 0 并且 a_p 小于 β，

$$q'_1 = q_1 + \mathrm{Clip3}(-t_{C0}, t_{C0}, (q_2 + ((p_0 + q_0 + 1) >> 1) - (q_1 << 1)) >> 1) \quad (8\text{-}339)$$

- Otherwise (chromaEdgeFlag is equal to 1 or a_q is greater than or equal to β),
- 否则（chromaEdgeFlag 等于 1 或者 a_p 大于等于 β，）

$$q'_1 = q_1 \quad (8\text{-}340)$$

The filtered result samples p'_2 and q'_2 are always set equal to the input samples p_2 and q_2:

滤波后的象素 p'_2 和 q'_2 总是设置成输入象素 p_2 和 q_2 的值。

$$p'_2 = p_2 \quad (8\text{-}341)$$

$$q'_2 = q_2 \quad (8\text{-}342)$$

Table 8-15 – Value of filter clipping variable t_{C0} as a function of indexA and bS

	indexA																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
bS = 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
bS = 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
bS = 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Table 8-15 (concluded) – Value of filter clipping variable t_{C0} as a function of indexA and bS

	indexA																									
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
bS = 1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13
bS = 2	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	6	7	8	8	10	11	12	13	15	17
bS = 3	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13	14	16	18	20	23	25

8.7.2.4 Filtering process for edges for bS equal to 4

8.7.2.4 对 bS 等于 4 的边缘的滤波过程

Inputs to this process are the input sample values p_i and q_i ($i = 0..3$) of a single set of samples across an edge that is to be filtered, the variable chromaEdgeFlag, and the values of the threshold variables α and β for the set of samples, as specified in subclause 8.7.2.

本过程的输入是横跨被滤波边缘的象素 p_i 和 q_i ($i = 0..3$)、变量 chromaEdgeFlag 和阈值变量 α 和 β，如 8.7.2 所示。

Outputs of this process are the filtered result sample values p'_i and q'_i ($i = 0..2$) for the set of input sample values.

本过程的输出是滤波后的象素值 p'_i 和 q'_i ($i = 0..2$)。

Let a_p and a_q be two threshold variables as specified in Equations 8-335 and 8-336, respectively, in subclause 8.7.2.3.

假设 a_p 和 a_q 是 8.7.2.3 小节中等式 8-335 和 8-336 分别说明的两个阈值。

The filtered result samples p'_i ($i = 0..2$) are derived as follows.

滤波后的象素 p'_i ($i = 0..2$) 计算如下：

- If chromaEdgeFlag is equal to 0 and the following condition holds,
- 如果 chromaEdgeFlag 等于 0 并且下面的条件满足，

$$a_p < \beta \ \&\& \ \text{Abs}(p_0 - q_0) < ((\alpha >> 2) + 2) \tag{8-343}$$

  then the variables p'_0, p'_1, and p'_2 are derived by

  然后变量 p'_0, p'_1, 和 p'_2 的计算如下，

$$p'_0 = (p_2 + 2*p_1 + 2*p_0 + 2*q_0 + q_1 + 4) >> 3 \tag{8-344}$$

$$p'_1 = (p_2 + p_1 + p_0 + q_0 + 2) >> 2 \tag{8-345}$$

$$p'_2 = (2*p_3 + 3*p_2 + p_1 + p_0 + q_0 + 4) >> 3 \tag{8-346}$$

- Otherwise (chromaEdgeFlag is equal to 1 or the condition in Equation 8-343 does not hold), the variables p'_0, p'_1, and p'_2 are derived by
- 否则（chromaEdgeFlag 等于 1 或者等式 8-343 中的条件成立），变量 p'_0, p'_1, 和 p'_2 的计算如下，

$$p'_0 = (2*p_1 + p_0 + q_1 + 2) >> 2 \quad (8\text{-}347)$$

$$p'_1 = p_1 \quad (8\text{-}348)$$

$$p'_2 = p_2 \quad (8\text{-}349)$$

The filtered result samples q'_i ($i = 0..2$) are derived as follows.

滤波后的象素 q'_i ($i = 0..2$) 的计算如下：

- If chromaEdgeFlag is equal to 0 and the following condition holds,
- 如果 chromaEdgeFlag 等于1 并且下面的条件成立，

$$a_q < \beta \ \ \&\& \ \ \mathrm{Abs}(p_0 - q_0) < ((\alpha >> 2) + 2) \quad (8\text{-}350)$$

then the variables q'_0, q'_1, and q'_2 are derived by

然后变量 p'_0, p'_1, 和 p'_2 的计算如下，

$$q'_0 = (p_1 + 2*p_0 + 2*q_0 + 2*q_1 + q_2 + 4) >> 3 \quad (8\text{-}351)$$

$$q'_1 = (p_0 + q_0 + q_1 + q_2 + 2) >> 2 \quad (8\text{-}352)$$

$$q'_2 = (2*q_3 + 3*q_2 + q_1 + q_0 + p_0 + 4) >> 3 \quad (8\text{-}353)$$

- Otherwise (chromaEdgeFlag is equal to 1 or the condition in Equation 8-350 does not hold), the variables q'_0, q'_1, and q'_2 are derived by
- 否则（chromaEdgeFlag 等于 1 或者等式 8-350 中的条件满足），变量 p'_0, p'_1, 和 p'_2 的计算如下，

$$q'_0 = (2*q_1 + q_0 + p_1 + 2) >> 2 \quad (8\text{-}354)$$

$$q'_1 = q_1 \quad (8\text{-}355)$$

$$q'_2 = q_2 \quad (8\text{-}356)$$

9 Parsing process

9 解析过程

Inputs to this process are bits from the RBSP.

本过程的输入是来自 RBSP 的比特

Outputs of this process are syntax element values.

本过程的输出是语法元素值

This process is invoked when the descriptor of a syntax element in the syntax tables in subclause 7.3 is equal to ue(v), me(v), se(v), te(v) (see subclause 9.1), ce(v) (see subclause 9.2), or ae(v) (see subclause 9.3).

在 7.3 小节的语法表中，当语法元素的描述符为 ue(v), me(v), se(v), te(v) (见 9.1 小节), ce(v) (见 9.2 小节), 或 ae(v) (见 9.3 小节) 时使用本过程。

9.1 Parsing process for Exp-Golomb codes

9.1 Exp-Golomb 码的解析过程

This process is invoked when the descriptor of a syntax element in the syntax tables in subclause 7.3 is equal to ue(v), me(v), se(v), or te(v). For syntax elements in subclauses 7.3.4 and 7.3.5, this process is invoked only when entropy_coding_mode_flag is equal to 0.

在 7.3 小节的语法表中，当语法元素的描述符为 ue(v), me(v), se(v), te(v) 时，使用本过程。对于 7.3.4 和 7.3.5 中的语法元素，只有当 entropy_coding_mode_flag 等于 0 时才使用本过程。

Inputs to this process are bits from the RBSP.

本过程的输入是来自 RBSP 的比特数据。

Outputs of this process are syntax elements.

本过程的输出是语法元素。

Syntax elements coded as ue(v), me(v), or se(v) are Exp-Golomb-coded. Syntax elements coded as te(v) are truncated Exp-Golomb-coded. The parsing process for these syntax elements begins with reading the bits starting at the current location in the bitstream up to and including the first non-zero bit, and counting the number of leading bits that are equal to 0. This process shall be equivalent to the following:

编码为 ue(v), me(v), 或者 se(v) 的语法元素为 Exp-Golomb 编码，编码为 te(v) 是截断 Exp-Golomb 编码。这些语法元素的解析过程首先从当前位置开始读取比特数据，直到第一个非零的比特为止，然后计算刚才读取的非零比特数据的个数。这个过程的描述如下：

```
leadingZeroBits = -1;
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )
```

The variable codeNum is then assigned as follows:

变量 codeNum 的赋值如下：

$$codeNum = 2^{leadingZeroBits} – 1 + read\_bits(leadingZeroBits)$$

where the value returned from read_bits(leadingZeroBits) is interpreted as a binary representation of an unsigned integer with most significant bit written first.

这里函数 read_bits(leadingZeroBits) 的返回值是无符号整数的二进制表示，首先是最重要的比特位。

Table 9-1 illustrates the structure of the Exp-Golomb code by separating the bit string into “prefix” and “suffix” bits. The “prefix” bits are those bits that are parsed in the above pseudo-code for the computation of leadingZeroBits, and are shown as either 0 or 1 in the bit string column of Table 9-1. The “suffix” bits are those bits that are parsed in the computation of codeNum and are shown as x_i in Table 9-1, with i being in the range 0 to leadingZeroBits - 1, inclusive. Each x_i can take on values 0 or 1.

表 Table 9-1 说明了 Exp-Golomb 码的结构，比特串被分割为“前缀”比特和“后缀”比特。前缀比特是在上面伪代码中用来计算 leadingZeroBits 的那些比特，如表 Table 9-1 比特串列中的 0 或 1 ，后缀比特是在计算 codeNum 时解析的那些比特，如表 Table 9-1 中的 x_i，i 的范围是 0 到 leadingZeroBits – 1，包括这两个数。x_i 的值可以为 0 或者 1 。

Table 9-1 – Bit strings with “prefix” and “suffix” bits and assignment to codeNum ranges (informative)

Bit string form	Range of codeNum
1	0
0 1 x_0	1-2
0 0 1 x_1 x_0	3-6
0 0 0 1 x_2 x_1 x_0	7-14
0 0 0 0 1 x_3 x_2 x_1 x_0	15-30
0 0 0 0 0 1 x_4 x_3 x_2 x_1 x_0	31-62
…	…

Table 9-2 illustrates explicitly the assignment of bit strings to codeNum values.

表 Table 9-2 说明了比特串到 codeNum 值的分配过程

Table 9-2 – Exp-Golomb bit strings and codeNum in explicit form and used as ue(v) (informative)

Bit string	codeNum

1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
…	…

Depending on the descriptor, the value of a syntax element is derived as follows.

根据描述符，一个语法元素值的计算如下：

- If the syntax element is coded as ue(v), the value of the syntax element is equal to codeNum.
- 如果语法元素编码为 ue(v)，那么语法元素的值等于 codeNum 。
- If the syntax element is coded as se(v), the value of the syntax element is derived by invoking the mapping process for signed Exp-Golomb codes as specified in subclause 9.1.1 with codeNum as the input.
- 如果语法元素编码为 se(v)，那么语法元素的值使用 9.1.1 小节中的有符号 Exp-Golomb 码的映射处理过程计算得到，在 9.1.1 小节中使用 codeNum 作为输入。
- If the syntax element is coded as me(v), the value of the syntax element is derived by invoking the mapping process for coded block pattern as specified in subclause 9.1.2 with codeNum as the input.
- 如果语法元素编码为 me(v) ，那么语法元素的值使用 9.1.2 小节中的编码块模式的映射处理过程计算得到，在 9.1.2 小节中使用 codeNum 作为输入。
- Otherwise (the syntax element is coded as te(v)), the range of the syntax element shall be determined first. The range of this syntax element may be between 0 and x, with x being greater than or equal to 1 and is used in the derivation of the value of a syntax element as follows
- 否则（语法元素编码成 te(v)），那么应该首先确定语法元素的范围，语法元素值的范围可能在 0 和 x 之间，这里 x 可能大于或者等于 1，在计算语法元素值的过程中使用如下：
  - If x is greater than 1, codeNum and the value of the syntax element shall be derived in the same way as for syntax elements coded as ue(v)
  - 如果 x 大于 1 ，那么 codeNum 和语法元素值的计算和 ue(v) 的语法元素相同。
  - Otherwise (x is equal to 1), the parsing process for codeNum which is equal to the value of the syntax element is given by a process equivalent to:
  - 否则（x 等于 1），那么 codeNum 等于语法元素的值，它的计算过程如下：

```
b = read_bits( 1 )
codeNum = !b
```

9.1.1 Mapping process for signed Exp-Golomb codes

9.1.1 有符号 Exp-Golomb 码的映射过程

Input to this process is codeNum as specified in subclause 9.1.

本过程的输入是 codeNum ， 如 9.1 小节所示。

Output of this process is a value of a syntax element coded as se(v).

本过程的输出是编码为 se(v) 的语法元素值。

The syntax element is assigned to the codeNum by ordering the syntax element by its absolute value in increasing order and representing the positive value for a given absolute value with the lower codeNum. Table 9-3 provides the assignment rule.

通过对语法元素的绝对值按照升序进行排列的方法将语法元素分配给 codeNum，并且绝对值的正值有较小的 codeNum，表 Table 9-3 表示了这样一个分配准则。

Table 9-3 – Assignment of syntax element to codeNum for signed Exp-Golomb coded syntax elements se(v)

codeNum	syntax element value
0	0
1	1
2	–1
3	2
4	–2
5	3
6	–3
k	$(-1)^{k+1}$ Ceil(k÷2)

9.1.2 Mapping process for coded block pattern

9.1.2 编码块模式的映射处理过程

Input to this process is codeNum as specified in subclause 9.1.

本过程的输入是 codeNum， 如 9.1 小节所示。

Output of this process is a value of the syntax element coded_block_pattern coded as me(v).

本过程的输出是编码为 me(v) 的语法元素 coded_block_pattern 的值。

Table 9-4 shows the assignment of coded_block_pattern to codeNum depending on whether the macroblock prediction mode is equal to Intra_4x4 or Inter.

表 Table 9-4 表示了根据宏块预测模式为 Intra_4x4 或者 Inter 的 coded_block_pattern 到 codeNum 的分配过程。

Table 9-4 – Assignment of codeNum to values of coded_block_pattern for macroblock prediction modes

codeNum	coded_block_pattern	
	Intra_4x4	Inter
0	47	0
1	31	16
2	15	1
3	0	2
4	23	4
5	27	8
6	29	32

7	30	3
8	7	5
9	11	10
10	13	12
11	14	15
12	39	47
13	43	7
14	45	11
15	46	13
16	16	14
17	3	6
18	5	9
19	10	31
20	12	35
21	19	37
22	21	42
23	26	44
24	28	33
25	35	34
26	37	36
27	42	40
28	44	39
29	1	43
30	2	45
31	4	46
32	8	17
33	17	18
34	18	20
35	20	24
36	24	19
37	6	21
38	9	26
39	22	28
40	25	23
41	32	27
42	33	29

43	34	30
44	36	22
45	40	25
46	38	38
47	41	41

9.2 CAVLC parsing process for transform coefficient levels

9.2 变换系数值的 CAVLC 解析过程

This process is invoked when parsing syntax elements with descriptor equal to ce(v) in subclause 7.3.5.3.1 and when entropy_coding_mode_flag is equal to 0.

当解析 7.3.5.3.1 小节中描述符为 ce(v) 的语法元素并且 entropy_coding_mode_flag 等于 0 的时候使用本过程。

Inputs to this process are bits from slice data, a maximum number of non-zero transform coefficient levels maxNumCoeff, the luma block index luma4x4BlkIdx or the chroma block index chroma4x4BlkIdx of the current block of transform coefficient levels.

本过程的输入来自 slice 的数据，非零变换系数值的最大数目 maxNumCoeff，当前块的亮度块索引 luma4x4BlkIdx 或者色差块索引 chroma4x4BlkIdx 。

Output of this process is the list coeffLevel containing transform coefficient levels of the luma block with block index luma4x4BlkIdx or the chroma block with block index chroma4x4BlkIdx.

本过程的输出是 coeffLevel 列表，包含有索引为 luma4x4BlkIdx 的亮度块或者索引为 chroma4x4BlkIdx 的色差块的变换系数值。

The process is specified in the following ordered steps:

本过程按照下面的步骤进行处理：

1. All transform coefficient levels, with indices from 0 to maxNumCoeff - 1, in the list coeffLevel are set equal to 0.

   在 coeffLevel 列表中索引为 0 到 maxNumCoeff – 1 的所有变换系数值都设置成 0；

2. The total number of non-zero transform coefficient levels TotalCoeff(coeff_token) and the number of trailing one transform coefficient levels TrailingOnes(coeff_token) are derived by parsing coeff_token (see subclause 9.2.1).

   通过解析 coeff_token 的值（见 9.2.1 小节），计算出非零变换系数值的总数 TotalCoeff(coeff_token) 和 TrailingOnes 的值 TrailingOnes(coeff_token) 。

   - If the number of non-zero transform coefficient levels TotalCoeff(coeff_token) is equal to 0, the list coeffLevel containing 0 values is returned and no further step is carried out.
   - 如果非零变换系数的个数 TotalCoeff(coeff_token) 为零，那么返回包含有零的 coeffLevel 列表，并且结束该块的解码。
   - Otherwise, the following steps are carried out.
   - 否则，执行下面的步骤：

     a. The non-zero transform coefficient levels are derived by parsing trailing_ones_sign_flag, level_prefix, and level_suffix (see subclause 9.2.2).

        通过解析 trailing_ones_sign_flag, level_prefix, 和 level_suffix （见 9.2.2 小节）来计算非零变换系数的值。

     b. The runs of zero transform coefficient levels before each non-zero transform coefficient level are derived by parsing total_zeros and run_before (see subclause 9.2.3).

        通过解析 total_zeros 和 run_before 来计算在每个非零变换系数之前零变换系数的行程（见 9.2.3 小节）。

c. The level and run information are combined into the list coeffLevel (see subclause 9.2.4).

值和行程信息被合并到 coeffLevel 列表中（见 9.2.4 小节）。

9.2.1 Parsing process for total number of transform coefficient levels and trailing ones

整个非零变换系数和 trailing ones 数目的解析过程

Inputs to this process are bits from slice data, a maximum number of non-zero transform coefficient levels maxNumCoeff, the luma block index luma4x4BlkIdx or the chroma block index chroma4x4BlkIdx of the current block of transform.

本过程的输入是来自 slice 层的码流数据，非零变换系数值的个数 maxNumCoeff，当前亮度块索引 luma4x4BlkIdx 或者色差块索引 chroma4x4BlkIdx 。

Outputs of this process are TotalCoeff(coeff_token) and TrailingOnes(coeff_token).

本过程的输出是 TotalCoeff(coeff_token) 和 TrailingOnes(coeff_token) 。

The syntax element coeff_token is decoded using one of the five VLCs specified in five right-most columns of Table 9-5. Each VLC specifies both TotalCoeff(coeff_token) and TrailingOnes(coeff_token) for a given codeword coeff_token. VLC selection is dependent upon a variable nC that is derived as follows.

使用表 Table 9-5 中最右边的五列 VLCs 来解码语法元素 coeff_token 。对于给定的码子 coeff_token ，每一个 VLC 都说明了 TotalCoeff(coeff_token) 和 TrailingOnes(coeff_token)，VLC 表的选择依赖于变量 nC，它的计算如下：

- If the CAVLC parsing process is invoked for ChromaDCLevel, nC is set equal to –1,
- 如果 CAVLC 的解析过程是在解码 ChromaDCLevel，那么 nC 的值被设置为 –1。
- Otherwise, the following applies.
- 否则，使用下面的过程
  - When the CAVLC parsing process is invoked for Intra16x16DCLevel, luma4x4BlkIdx is set equal to 0.
  - 当 CAVLC 的解析过程是在解码 Intra16x16DCLevel，那么 luma4x4BlkIdx 被设置成 0 。
  - The variables blkA and blkB are derived as follows.
  - 变量 blkA 和 blkB 的计算如下：
    - If the CAVLC parsing process is invoked for Intra16x16DCLevel, Intra16x16ACLevel, or LumaLevel, the process specified in subclause 6.4.7.3 is invoked with luma4x4BlkIdx as the input, and the output is assigned to mbAddrA, mbAddrB, luma4x4BlkIdxA, and luma4x4BlkIdxB. The 4x4 luma block specified by mbAddrA\luma4x4BlkIdxA is assigned to blkA, and the 4x4 luma block specified by mbAddrB\luma4x4BlkIdxB is assigned to blkB.
    - 如果 CAVLC 的解析过程是在解码 Intra16x16DCLevel，Intra16x16ACLevel 或者 LumaLevel，那么将 luma4x4BlkIdx 作为输入使用 6.4.7.3 小节中的处理过程，输出 mbAddrA, mbAddrB, luma4x4BlkIdxA, 和 luma4x4BlkIdxB 。mbAddrA\luma4x4BlkIdxA 表示的 4x4 亮度块赋给 blkA，mbAddrB\luma4x4BlkIdxB 表示 4x4 的亮度块赋给 blkB 。
    - Otherwise (the CAVLC parsing process is invoked for ChromaACLevel), the process specified in subclause 6.4.7.4 is invoked with chroma4x4BlkIdx as input, and the output is assigned to mbAddrA, mbAddrB, chroma4x4BlkIdxA, and chroma4x4BlkIdxB. The 4x4 chroma block specified by mbAddrA\iCbCr\chroma4x4BlkIdxA is assigned to blkA, and the 4x4 chroma block specified by mbAddrB\iCbCr\luma4x4BlkIdxB is assigned to blkB.
    - 否则（CAVLC 的解析过程是在解码 ChromaACLevel），那么将 chroma4x4BlkIdx 作为输入，使用 6.4.7.4 小节中的处理过程，输出赋给 mbAddrA, mbAddrB, chroma4x4BlkIdxA, 和 chroma4x4BlkIdxB，mbAddrA\iCbCr\chroma4x4BlkIdxA 表示的 4x4 色差块赋给 blkA，mbAddrB\iCbCr\luma4x4BlkIdxB 表示的 4x4色差块赋给 blkB 。
  - Let nA and nB be the number of non-zero transform coefficient levels (given by TotalCoeff(coeff_token)) in the block of transform coefficient levels blkA located to the left of the current block and the block of transform coefficient levels blkB located above the current block, respectively.

- 假设位于当前块左边的块的变换系数中，非零变换系数值的个数（由 TotalCoeff(coeff_token) 计算得到）为 nA ，位于当前块上边的块的变换系数中，非零变换系数值的个数（由 TotalCoeff(coeff_token) 计算得到）为 nB。
- With N replaced by A and B, in mbAddrN, blkN, and nN the following applies.
- 在 mbAddrN, blkN, 和 nN 中，N 可能是 A 或者 B ，使用下面的过程。
  - If any of the following conditions is true, nN is set equal to 0.
  - 如果下面的任何一个条件满足，那么 nN 被设置成 0。
    - mbAddrN is not available
    - mbAddrN 不可用
    - The current macroblock is coded using an Intra prediction mode, constrained_intra_pred_flag is equal to 1 and mbAddrN is coded using Inter prediction and slice data partitioning is in use (nal_unit_type is in the range of 2 to 4, inclusive).
    - 当前宏块是帧内编码模式，constrained_intra_pred_flag 等于 1，而 mbAddrN 是帧间预测模式，并且使用 slice 数据分割（nal_unit_type 的范围是 2 到 4 ，包括 2 和 4）。
    - The macroblock mbAddrN has mb_type equal to P_Skip or B_Skip
    - 宏块 mbAddrN 的类型为 P_Skip 或者 B_Skip。
    - All AC residual transform coefficient levels of the neighbouring block blkN are equal to 0 due to the corresponding bit of CodedBlockPatternLuma or CodedBlockPatternChroma being equal to 0
    - 由于 CodedBlockPatternLuma 或者 CodedBlockPatternChroma 的相应比特位为 0 ，相邻块 blkN 的所有 AC 变换系数都为 0 。
  - If mbAddrN is an I_PCM macroblock, nN is set equal to 16.
  - 如果 mbAddrN 的编码模式为 I_PCM ，那么 nN 被设置成 16 。
  - Otherwise, nN is set equal to the value TotalCoeff(coeff_token) of the neighbouring block blkN.
  - 否则，nN 被设置成相邻块 blkN 的 TotalCoeff(coeff_token) 。

    NOTE - The values nA and nB that are derived using TotalCoeff(coeff_token) do not include the DC transform coefficient levels in Intra 16x16 macroblocks or DC transform coefficient levels in chroma blocks, because these transform coefficient levels are decoded separately. When the block above or to the left belongs to an Intra 16x16 macroblock, or is a chroma block, nA and nB is the number of decoded non-zero AC transform coefficient levels.

    NOTE - When parsing for Intra16x16DCLevel, the values nA and nB are based on the number of non-zero transform coefficient levels in adjacent 4x4 blocks and not on the number of non-zero DC transform coefficient levels in adjacent 16x16 blocks.

    注一 使用 TotalCoeff(coeff_token) 计算得到的 nA 和 nB 在帧内编码宏块和色差块中不包含 DC 系数，因为这些系数是单独解码，当上面的块或者左边的块属于 16x16 帧内编码宏块或者色差块，那么 nA 和 nB 表示的是非零 AC 系数的个数。

    注一 当解析 Intra16x16DCLevel 时，nA 和 nB 的值是基于相邻 4x4 块的非零变换系数的个数，而不是基于相邻 16x16 块的非零 DC 变换系数的个数。

- Given the values of nA and nB, the variable nC is derived as follows.
- 根据 nA 和 nB 的值，变量 nC 的计算如下：
  - If both mbAddrA and mbAddrB are available, the variable nC is set equal to (nA + nB + 1) >> 1.
  - 如果 mbAddrA 和 mbAddrB 都是可用的，那么变量 nC 被设置成 (nA + nB + 1) >> 1。
  - Otherwise (mbAddrA is not available or mbAddrB is not available), the variable nC is set equal to nA + nB.
  - 否则（mbAddrA 不可用或者 mbAddrB 不可用），那么 变量 nC 被设置成 nA + nB。

The value of TotalCoeff(coeff_token) resulting from decoding coeff_token shall be in the range of 0 to maxNumCoeff, inclusive.

解码 coeff_token 得到的 TotalCoeff(coeff_token) 应该在 0 到 maxNumCoeff 的范围内。

Table 9-5 – coeff_token mapping to TotalCoeff(coeff_token) and TrailingOnes(coeff_token)

TrailingOnes (coeff_token)	TotalCoeff (coeff_token)	0 <= nC < 2	2 <= nC < 4	4 <= nC < 8	8 <= nC	nC = = -1
0	0	1	11	1111	0000 11	01
0	1	0001 01	0010 11	0011 11	0000 00	0001 11
1	1	01	10	1110	0000 01	1
0	2	0000 0111	0001 11	0010 11	0001 00	0001 00
1	2	0001 00	0011 1	0111 1	0001 01	0001 10
2	2	001	011	1101	0001 10	001
0	3	0000 0011 1	0000 111	0010 00	0010 00	0000 11
1	3	0000 0110	0010 10	0110 0	0010 01	0000 011
2	3	0000 101	0010 01	0111 0	0010 10	0000 010
3	3	0001 1	0101	1100	0010 11	0001 01
0	4	0000 0001 11	0000 0111	0001 111	0011 00	0000 10
1	4	0000 0011 0	0001 10	0101 0	0011 01	0000 0011
2	4	0000 0101	0001 01	0101 1	0011 10	0000 0010
3	4	0000 11	0100	1011	0011 11	0000 000
0	5	0000 0000 111	0000 0100	0001 011	0100 00	-
1	5	0000 0001 10	0000 110	0100 0	0100 01	-
2	5	0000 0010 1	0000 101	0100 1	0100 10	-
3	5	0000 100	0011 0	1010	0100 11	-
0	6	0000 0000 0111 1	0000 0011 1	0001 001	0101 00	-
1	6	0000 0000 110	0000 0110	0011 10	0101 01	-
2	6	0000 0001 01	0000 0101	0011 01	0101 10	-
3	6	0000 0100	0010 00	1001	0101 11	-
0	7	0000 0000 0101 1	0000 0001 111	0001 000	0110 00	-
1	7	0000 0000 0111 0	0000 0011 0	0010 10	0110 01	-
2	7	0000 0000 101	0000 0010 1	0010 01	0110 10	-
3	7	0000 0010 0	0001 00	1000	0110 11	-
0	8	0000 0000 0100 0	0000 0001 011	0000 1111	0111 00	-
1	8	0000 0000 0101 0	0000 0001 110	0001 110	0111 01	-
2	8	0000 0000 0110 1	0000 0001 101	0001 101	0111 10	-
3	8	0000 0001 00	0000 100	0110 1	0111 11	-
0	9	0000 0000 0011 11	0000 0000 1111	0000 1011	1000 00	-
1	9	0000 0000 0011 10	0000 0001 010	0000 1110	1000 01	-
2	9	0000 0000 0100 1	0000 0001 001	0001 010	1000 10	-

3	9	0000 0000 100	0000 0010 0	0011 00	1000 11	-
0	10	0000 0000 0010 11	0000 0000 1011	0000 0111 1	1001 00	-
1	10	0000 0000 0010 10	0000 0000 1110	0000 1010	1001 01	-
2	10	0000 0000 0011 01	0000 0000 1101	0000 1101	1001 10	-
3	10	0000 0000 0110 0	0000 0001 100	0001 100	1001 11	-
0	11	0000 0000 0001 111	0000 0000 1000	0000 0101 1	1010 00	-
1	11	0000 0000 0001 110	0000 0000 1010	0000 0111 0	1010 01	-
2	11	0000 0000 0010 01	0000 0000 1001	0000 1001	1010 10	-
3	11	0000 0000 0011 00	0000 0001 000	0000 1100	1010 11	-
0	12	0000 0000 0001 011	0000 0000 0111 1	0000 0100 0	1011 00	-
1	12	0000 0000 0001 010	0000 0000 0111 0	0000 0101 0	1011 01	-
2	12	0000 0000 0001 101	0000 0000 0110 1	0000 0110 1	1011 10	-
3	12	0000 0000 0010 00	0000 0000 1100	0000 1000	1011 11	-
0	13	0000 0000 0000 1111	0000 0000 0101 1	0000 0011 01	1100 00	-
1	13	0000 0000 0000 001	0000 0000 0101 0	0000 0011 1	1100 01	-
2	13	0000 0000 0001 001	0000 0000 0100 1	0000 0100 1	1100 10	-
3	13	0000 0000 0001 100	0000 0000 0110 0	0000 0110 0	1100 11	-
0	14	0000 0000 0000 1011	0000 0000 0011 1	0000 0010 01	1101 00	-
1	14	0000 0000 0000 1110	0000 0000 0010 11	0000 0011 00	1101 01	-
2	14	0000 0000 0000 1101	0000 0000 0011 0	0000 0010 11	1101 10	-
3	14	0000 0000 0001 000	0000 0000 0100 0	0000 0010 10	1101 11	-
0	15	0000 0000 0000 0111	0000 0000 0010 01	0000 0001 01	1110 00	-
1	15	0000 0000 0000 1010	0000 0000 0010 00	0000 0010 00	1110 01	-
2	15	0000 0000 0000 1001	0000 0000 0010 10	0000 0001 11	1110 10	-
3	15	0000 0000 0000 1100	0000 0000 0000 1	0000 0001 10	1110 11	-
0	16	0000 0000 0000 0100	0000 0000 0001 11	0000 0000 01	1111 00	-
1	16	0000 0000 0000 0110	0000 0000 0001 10	0000 0001 00	1111 01	-
2	16	0000 0000 0000 0101	0000 0000 0001 01	0000 0000 11	1111 10	-
3	16	0000 0000 0000 1000	0000 0000 0001 00	0000 0000 10	1111 11	-

9.2.2 Parsing process for level information

9.2.2 level 信息的解析过程

Inputs to this process are bits from slice data, the number of non-zero transform coefficient levels TotalCoeff(coeff_token), and the number of trailing one transform coefficient levels TrailingOnes(coeff_token).

本过程的输入是来自 slice 的数据，非零变换系数的个数 TotalCoeff(coeff_token)， 和 trailingone 变换系数的个数 TrailingOnes(coeff_token)。

Output of this process is a list with name level containing transform coefficient levels.

本过程的输出是包含变换系数值的列表，名称为 level。

Initially an index i is set equal to 0. Then the following procedure is iteratively applied TrailingOnes(coeff_token) times to decode the trailing one transform coefficient levels (if any):

刚开始索引 i 被初始化成 0 ，然后使用下面的过程 TrailingOnes(coeff_token) 次来解码 trailingone 变换系数值：

- A 1-bit syntax element trailing_ones_sign_flag is decoded and evaluated as follows.
- 解码一比特的语法元素 trailing_ones_sign_flag ，并且计算如下：
    - If trailing_ones_sign_flag is equal to 0, the value +1 is assigned to level[i].
    - 如果 trailing_ones_sign_flag 等于 0 ， 那么 +1 赋给 level[i]。
    - Otherwise (trailing_ones_sign_flag is equal to 1), the value -1 is assigned to level[i].
    - 否则（trailing_ones_sign_flag 等于 1），那么 –1 赋给 level[i]。
- The index i is incremented by 1.
- 索引增加 1 。

Following the decoding of the trailing one transform coefficient levels, a variable suffixLength is initialised as follows.

解码 trailingone 变换系数值之后，变量 suffixLength 的初始化如下：

- If TotalCoeff(coeff_token) is larger than 10 and TrailingOnes(coeff_token) is less than 3, suffixLength is set equal to 1.
- 如果 TotalCoeff(coeff_token) 大于 10 ，并且 TrailingOnes(coeff_token) 小于 3 ，那么 suffixLength 被设置成 1。
- Otherwise (TotalCoeff(coeff_token) is less than or equal to 10 or TrailingOnes(coeff_token) is equal to 3), suffixLength is set equal to 0.
- 否则（TotalCoeff(coeff_token) 小于 10 ，或者 TrailingOnes(coeff_token) 大于 3），那么 suffixLength 被设置成 0。

The following procedure is then applied iteratively (TotalCoeff(coeff_token) – TrailingOnes(coeff_token)) times to decode the remaining levels (if any):

然后使用下面的过程 (TotalCoeff(coeff_token) – TrailingOnes(coeff_token)) 次来解码剩余的变换系数的值。

- The syntax element level_prefix is decoded using the VLC specified in Table 9-6.
- 使用表 Table 9-6 来解码语法元素 level_prefix 。
- The variable levelSuffixSize is set equal to the variable suffixLength with the exception of the following two cases.
- 除了下面两种情况，变量 levelSuffixSize 被设置成变量 suffixLength。
- When level_prefix is equal to 14 and suffixLength is equal to 0, levelSuffixSize is set equal to 4.
- 当 level_prefix 等于 14 并且 suffixLength 等于 0，那么 levelSuffixSize 被设置成 4。
- When level_prefix is equal to 15, levelSuffixSize is set equal to 12.
- 当 level_prefix 等于 15，levelSuffixSize 被设置成 12。
- The syntax element level_suffix is decoded as follows.
- 语法元素 level_suffix 的解码如下：
    - If levelSuffixSize is greater than 0, the syntax element level_suffix is decoded as unsigned integer representation u(v) with levelSuffixSize bits.
    - 如果 levelSuffixSize 大于 0 ，语法元素 level_suffix 解码成一个无符号整数表示 u(v) ，位宽为 levelSuffixSize 比特。
    - Otherwise (levelSuffixSize is equal to 0), the syntax element level_suffix shall be inferred to be equal to 0.
    - 否则（levelSuffixSize 等于 0），那么语法元素 level_suffix 的值被参考成 0。

- A variable levelCode is set equal to (level_prefix << suffixLength) + level_suffix.
- 变量 levelCode 被设置成 (level_prefix << suffixLength) + level_suffix.
- When level_prefix is equal to 15 and suffixLength is equal to 0, levelCode is incremented by 15.
- 当 level_prefix 等于 15 并且 suffixLength 等于 0 时,那么 levelCode 的值增加 15;
- When the index i is equal to TrailingOnes(coeff_token) and TrailingOnes(coeff_token) is smaller than 3, levelCode is incremented by 2.
- 当索引 i 等于 TrailingOnes(coeff_token) 并且 TrailingOnes(coeff_token) 的值小于 3 时, levelCode 的值增加 2 ;
- The variable level[i] is derived as follows.
- 变量 level[i] 的计算如下:
  - If levelCode is an even number, the value (levelCode + 2) >> 1 is assigned to level[i].
  - Otherwise, the value (-levelCode – 1) >> 1 is assigned to level[i].
  - 如果 levelCode 为偶数,那么 (levelCode + 2) >> 1 的值赋给 level[i];
  - 否则 (-levelCode – 1) >> 1 赋给 level[i];
- When suffixLength is equal to 0, suffixLength is set equal to 1.
- 当 suffixLength 的值等于 0 时,那么 suffixLength 被设置成 1 ;
- When the absolute value of level[i] is larger than (3 << (suffixLength – 1)) and suffixLength is less than 6, suffixLength is incremented by 1.
- 当 level[i] 的绝对值大于 (3 << (suffixLength – 1)) 并且 suffixLength 小于 6 时,那么 suffixLength 的值增加 1;
- The index i is incremented by 1.
- 索引 i 增加 1;

Table 9-6 – Codeword table for level_prefix

level_prefix	bit string
0	1
1	01
2	001
3	0001
4	0000 1
5	0000 01
6	0000 001
7	0000 0001
8	0000 0000 1
9	0000 0000 01
10	0000 0000 001
11	0000 0000 0001
12	0000 0000 0000 1
13	0000 0000 0000 01
14	0000 0000 0000 001
15	0000 0000 0000 0001

9.2.3 Parsing process for run information

9.2.3 run 信息的解析过程

Inputs to this process are bits from slice data, the number of non-zero transform coefficient levels TotalCoeff(coeff_token), and the maximum number of non-zero transform coefficient levels maxNumCoeff.

本过程的输入是来自 slice 数据的比特位,非零变换系数值的数目 TotalCoeff(coeff_token),和非零变换系数值的最大数目 maxNumCoeff.

Output of this process is a list of runs of zero transform coefficient levels preceding non-zero transform coefficient levels called run.

本过程的输出是非零变换系数值之前零值变换系数的行程列表,称为 run .

Initially, an index i is set equal to 0.

首先,索引 i 被设置成 0.

The variable zerosLeft is derived as follows.

变量 zerosLeft 的计算如下:

- If the number of non-zero transform coefficient levels TotalCoeff(coeff_token) is equal to the maximum number of non-zero transform coefficient levels maxNumCoeff, a variable zerosLeft is set equal to 0.
- 如果非零变换系数的个数 TotalCoeff(coeff_token) 等于最大非零变换系数值的个数 maxNumCoeff ,那么变量 zerosLeft 的值被设置成 0;
- Otherwise (the number of non-zero transform coefficient levels TotalCoeff(coeff_token) is less than the maximum number of non-zero transform coefficient levels maxNumCoeff), total_zeros is decoded and zerosLeft is set equal to its value.
- 否则(非零变换系数的个数 TotalCoeff(coeff_token) 小于最大非零变换系数值的个数 maxNumCoeff),那么解码 total_zeros 并且 zerosLeft 被设置成对应的值.

The VLC used to decode total_zeros is derived as follows:

用于解码 total_zeros 的 VLC 码表计算如下:

- If maxNumCoeff is equal to 4 one of the VLCs specified in Table 9-9 is used.
- Otherwise (maxNumCoeff is not equal to 4), VLCs from Table 9-7 and Table 9-8 are used.
- 如果 maxNumCoeff 的值等于 4 ,那么使用 Table 9-9 说明的一个 VLC 表;
- 否则(maxNumCoeff 的值不等于 4),那么使用 Table 9-7 和 Table 9-8 中的 VLC 表;

The following procedure is then applied iteratively (TotalCoeff(coeff_token) – 1) times:

然后使用下面的过程 (TotalCoeff(coeff_token) – 1) 次:

- The variable run[i] is derived as follows.
- 变量 run[i] 的计算如下:
  - If zerosLeft is larger than zero, a value run_before is decoded based on Table 9-10 and zerosLeft. run[i] is set equal to run_before.
  - Otherwise (zerosLeft is equal to 0), run[i] is set equal to 0.
  - 如果 zerosLeft 大于零,那么基于表 Table 9-10 和 zerosLeft 的值解码 run_before , run[i] 也被设置成 run_before.
  - 否则(zerosLeft 等于零), run[i] 则被设置成 0;
- The value of run[i] is subtracted from zerosLeft and the result assigned to zerosLeft. The result of the subtraction shall be larger than or equal to 0.
- The index i is incremented by 1.

- 然后从 zerosLeft 中减去 run[i] 并将结果赋给 zerosLeft,结果应该大于或者等于 0 ;
- 索引 i 增加 1;

Finally the value of zerosLeft is assigned to run[i].

最后 zerosLeft 的值赋给 run[i] .

Table 9-7 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 1 to 7

total_zeros	**TotalCoeff(coeff_token)**						
	1	2	3	4	5	6	7
0	1	111	0101	0001 1	0101	0000 01	0000 01
1	011	110	111	111	0100	0000 1	0000 1
2	010	101	110	0101	0011	111	101
3	0011	100	101	0100	111	110	100
4	0010	011	0100	110	110	101	011
5	0001 1	0101	0011	101	101	100	11
6	0001 0	0100	100	100	100	011	010
7	0000 11	0011	011	0011	011	010	0001
8	0000 10	0010	0010	011	0010	0001	001
9	0000 011	0001 1	0001 1	0010	0000 1	001	0000 00
10	0000 010	0001 0	0001 0	0001 0	0001	0000 00	
11	0000 0011	0000 11	0000 01	0000 1	0000 0		
12	0000 0010	0000 10	0000 1	0000 0			
13	0000 0001 1	0000 01	0000 00				
14	0000 0001 0	0000 00					
15	0000 0000 1						

Table 9-8 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 8 to 15

total_zeros	**TotalCoeff(coeff_token)**							
	8	9	10	11	12	13	14	15
0	0000 01	0000 01	0000 1	0000	0000	000	00	0
1	0001	0000 00	0000 0	0001	0001	001	01	1
2	0000 1	0001	001	001	01	1	1	
3	011	11	11	010	1	01		
4	11	10	10	1	001			
5	10	001	01	011				
6	010	01	0001					
7	001	0000 1						

8	0000 00							

Table 9-9 – total_zeros tables for chroma DC 2x2 blocks

total_zeros	TotalCoeff(coeff_token)		
	1	2	3
0	1	1	1
1	01	01	0
2	001	00	
3	000		

Table 9-10 – Tables for run_before

run_before	zerosLeft						
	1	2	3	4	5	6	>6
0	1	1	11	11	11	11	111
1	0	01	10	10	10	000	110
2	-	00	01	01	011	001	101
3	-	-	00	001	010	011	100
4	-	-	-	000	001	010	011
5	-	-	-	-	000	101	010
6	-	-	-	-	-	100	001
7	-	-	-	-	-	-	0001
8		-	-	-	-	-	00001
9	-	-	-	-	-	-	000001
10	-	-	-	-	-	-	0000001
11	-	-	-	-	-	-	00000001
12	-	-	-	-	-	-	000000001
13	-	-	-	-	-	-	0000000001
14	-	-	-	-	-	-	00000000001

9.2.4 Combining level and run information

9.2.4 level 和 run 信息的合并

Input to this process are a list of transform coefficient levels called level, a list of runs called run, and the number of non-zero transform coefficient levels TotalCoeff(coeff_token).

本过程的输入是变换值列表 level 和行程列表 run 和非零变换系数值的数目 TotalCoeff(coeff_token) .

Output of this process is an list coeffLevel of transform coefficient levels.

本过程的输出是变换系数值列表 coeffLevel .

A variable coeffNum is set equal to -1 and an index i is set equal to (TotalCoeff(coeff_token) – 1). The following procedure is iteratively applied TotalCoeff(coeff_token) times:

变量 coeffNum 被设置成 -1 ,并且索引 i 被设置成 (TotalCoeff(coeff_token) – 1) ,然后使用下面的过程 TotalCoeff(coeff_token) 次:

- coeffNum is incremented by run[i] + 1.
- coeffLevel[coeffNum] is set equal to level[i].
- The index i is decremented by 1.
- coeffNum 增加 run[i] + 1 ;
- coeffLevel[coeffNum] 被设置成 level[i] ;
- 索引 i 减 1 ;

9.3 CABAC parsing process for slice data

9.3 slice 数据的 CABAC 解析过程

This process is invoked when parsing syntax elements with descriptor ae(v) in subclauses 7.3.4 and 7.3.5 when entropy_coding_mode_flag is equal to 1.

对于 entropy_coding_mode_flag 等于 1 并且在 7.3.4 和 7.3.5 小节中的描述符为 ae(v) 的语法元素使用本过程。

Inputs to this process are a request for a value of a syntax element and values of prior parsed syntax elements.

本过程的输入是：当前语法元素的解码需求和前面已经解析的语法元素的值。

Output of this process is the value of the syntax element.

本过程的输出是当前语法元素的值。

When starting the parsing of the slice data of a slice in subclause 7.3.4, the initialisation process of the CABAC parsing process is invoked as specified in subclause 9.3.1.

当开始解码 7.3.4 小节中的 slice 数据时，首先使用 9.3.1 小节中的 CABAC 的初始化过程。

The parsing of syntax elements proceeds as follows:

语法元素的解析过程按下面的步骤进行：

For each requested value of a syntax element a binarization is derived as described in subclause 9.3.2.

对于每一个需要解码的语法元素来说，先调用 9.3.2 小节中描述的二值化过程。

The binarization for the syntax element and the sequence of parsed bins determines the decoding process flow as described in subclause 9.3.3.

语法元素的二值化过程和解析的二进制序列决定了整个解码过程，如 9.3.3 小节所示。

For each bin of the binarization of the syntax element, which is indexed by the variable binIdx, a context index ctxIdx is derived as specified in subclause 9.3.3.1.

对于语法元素二值化后产生的二进制位，它的索引为 binIdx ，根据 9.3.3.1 小节的描述计算出相应的内容模型索引 ctxIdx 。

For each ctxIdx the arithmetic decoding process is invoked as specified in subclause 9.3.3.2.

对于索引为 ctxIdx 的内容模型，再使用 9.3.3.2 小节中的解码过程。

The resulting sequence (b_0 .. b_{binIdx}) of parsed bins is compared to the set of bin strings given by the binarization process after decoding of each bin. When the sequence matches a bin string in the given set, the corresponding value shall be assigned to the syntax element.

解码完二进制位后，解析得到的二进制序列 (b_0 .. b_{binIdx}) 根据二值化的过程和二进制串集进行比较，当这个序列和一个二进制串相互匹配，那么对应的值就赋给解码的语法元素。

In case the request for a value of a syntax element is processed for the syntax element mb_type and the decoded value of mb_type is I_PCM, the decoding engine shall be initialised after the decoding of the pcm_alignment_zero_bit and all pcm_byte data as specified in subclause 9.3.1.2.

当解码的语法元素为 mb_type 并且解码得到的值为 I_PCM ，那么在 pcm_alignment_zero_bit 和所有的 pcm_byte 数据解码之后，如 9.3.1.2 小节所示，需要初始化解码引擎。

The whole CABAC parsing process is illustrated in the flowchart of Figure 9-1 with the abbreviation SE for syntax element.

整个 CABAC 的解析过程如图 Figure 9-1 所示，这里 SE 是语法元素的缩写。

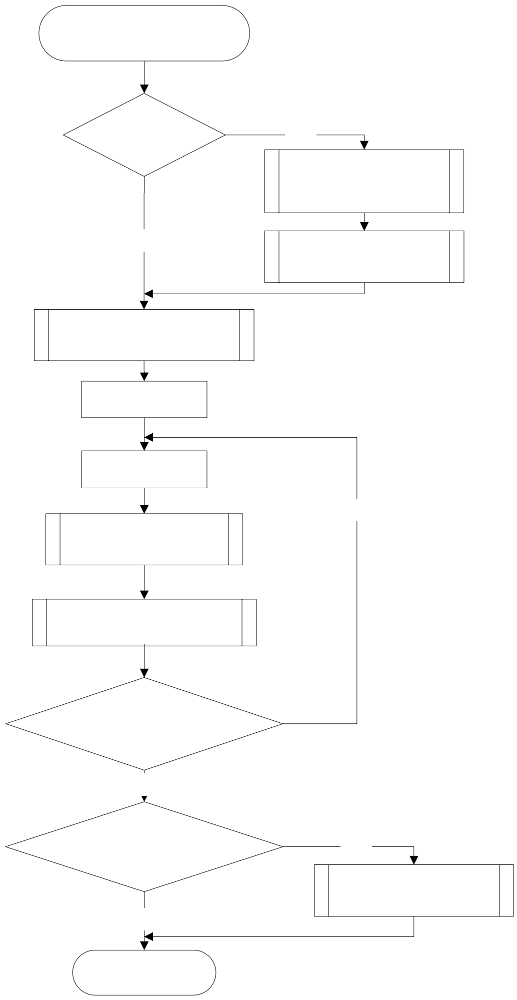


Figure 9-1 – Illustration of CABAC parsing process for a syntax element SE (informative)

9.3.1 Initialisation process

9.3.1 初始化的过程

Outputs of this process are initialised CABAC internal variables.

本过程的输出是初始化后的 CABAC 内部变量。

The processes in subclauses 9.3.1.1 and 9.3.1.2 are invoked when starting the parsing of the slice data of a slice in subclause 7.3.4.

当开始解析 7.3.4 小节中的 slice 数据时使用 9.3.1.1 和 9.3.1.2 小节中的处理过程。

The process in subclause 9.3.1.2 is also invoked after decoding the pcm_alignment_zero_bit and all pcm_byte data for a macroblock of type I_PCM.

对于类型为 I_PCM 的宏块来说，解码完 pcm_alignment_zero_bit 和所有 pcm_byte 数据之后再调用 9.3.1.2 小节中的处理过程。

9.3.1.1 Initialisation process for context variables

9.3.1.1 内容变量的初始化

Outputs of this process are the initialised CABAC context variables indexed by ctxIdx.

本过程的输出是：初始化后的索引为 ctxIdx 的 CABAC 内容变量。

Table 9-12 to Table 9-23 contain the values of the variables n and m used in the initialisation of context variables that are assigned to all syntax elements in subclauses 7.3.4 and 7.3.5 except for the end-of-slice flag.

表 Table 9-12 到 Table 9-23 包含了初始化内容模型的过程中使用的变量 m 和 n ，这些内容模型分别分配给 7.3.4 和 7.3.5 小节中的语法元素，除 end-of-slice 标志位以外。

For each context variable, the two variables pStateIdx and valMPS are initialised.

对于每一个内容模型，两个变量 pStateIdx 和 valMPS 需要初始化。

NOTE - The variable pStateIdx corresponds to a probability state index and the variable valMPS corresponds to the value of the most probable symbol as further described in subclause 9.3.3.2.

注意：变量 pStateIdx 对应了概率状态索引，变量 valMPS 对应了大概率符号（most probable symbol）的值，如 9.3.3.2 小节的描述。

The two values assigned to pStateIdx and valMPS for the initialisation are derived from $SliceQP_Y$, which is derived in Equation 7-16. Given the two table entries (m, n),

初始化过程中分配给 pStateIdx 和 valMPS 的两个值由 $SliceQP_Y$ 计算得到，$SliceQP_Y$ 的计算如等式 7-16 所示，根据给定两个表的索引（m，n）：

1. preCtxState = Clip3(1, 126, ((m ∗ $SliceQP_Y$) >> 4) + n)
2. if(preCtxState <= 63) {

   pStateIdx = 63 - preCtxState

   valMPS = 0

   } else {

   pStateIdx = preCtxState - 64

   valMPS = 1

   }

In Table 9-11, the ctxIdx for which initialisation is needed for each of the slice types are listed. Also listed is the table number that includes the values of m and n needed for the initialisation. For P, SP and B slice type, the initialisation depends also on the value of the cabac_init_idc syntax element. Note that the syntax element names do not affect the initialisation process.

在 Table 9-11 中，列出了对于每一种 slice 类型需要初始化的内容模型的索引 ctxIdx，同时也列出了检索初始化过程中使用 m 和 n 值的表号。对于 P, SP 和 B slice 类型，初始化的过程还依赖于语法元素 cabac_init_idc 的值，注意语法元素的名称不影响初始化的过程。

Table 9-11 – Association of ctxIdx and syntax elements for each slice type in the initialisation process

	Syntax element	Table	Slice type			
			SI	I	P, SP	B
slice_data( )	mb_skip_flag	Table 9-13 Table 9-14			11-13	24-26
	mb_field_decoding_flag	Table 9-18	70-72	70-72	70-72	70-72
macroblock_layer( )	mb_type	Table 9-12, Table 9-13, Table 9-14.	0-10	3-10	14-20	27-35

	coded_block_pattern (luma)	Table 9-18	73-76	73-76	73-76	73-76
	coded_block_pattern (chroma)	Table 9-18	77-84	77-84	77-84	77-84
	mb_qp_delta	Table 9-17	60-63	60-63	60-63	60-63
mb_pred( )	prev_intra4x4_pred_mode_flag	Table 9-17	68	68	68	68
	rem_intra4x4_pred_mode	Table 9-17	69	69	69	69
	intra_chroma_pred_mode	Table 9-17	64-67	64-67	64-67	64-67
mb_pred( ) and sub_mb_pred( )	ref_idx_l0	Table 9-16			54-59	54-59
	ref_idx_l1	Table 9-16				54-59
	mvd_l0[ ][ ][0]	Table 9-15			40-46	40-46
	mvd_l1[ ][ ][0]	Table 9-15				40-46
	mvd_l0[ ][ ][1]	Table 9-15			47-53	47-53
	mvd_l1[ ][ ][1]	Table 9-15				47-53
sub_mb_pred( )	sub_mb_type	Table 9-13 Table 9-14			21-23	36-39
residual_block_cabac( )	coded_block_flag	Table 9-18	85-104	85-104	85-104	85-104
	significant_coeff_flag[ ]	Table 9-19, Table 9-22.	105-165, 277-337	105-165, 277-337	105-165, 277-337	105-165, 277-337
	last_significant_coeff_flag[ ]	Table 9-20, Table 9-23.	166-226, 338-398	166-226, 338-398	166-226, 338-398	166-226, 338-398
	coeff_abs_level_minus1[ ]	Table 9-21	227-275	227-275	227-275	227-275

NOTE – ctxIdx equal to 276 is associated with the end_of_slice_flag and the bin of mb_type, which specifies the I_PCM macroblock type. The decoding process specified in subclause 9.3.3.2.4 applies to ctxIdx equal to 276. This decoding process, however, may also be implemented by using the decoding process specified in subclause 9.3.3.2.1. In this case, the initial values associated with ctxIdx equal to 276 are specified to be pStateIdx = 63 and valMPS = 0, where pStateIdx = 63 represents a non-adapting probability state.

注意：ctxIdx 等于 276 的内容模型与 end_of_slice_flag 以及类型为 I_PCM 的宏块的 mb_type 的二进制有关。ctxIdx 等于 276 的解码过程如 9.3.3.2.4 所示，但是解码过程的实现还是采用的 9.3.3.2.1 小节中的方法。在这种情况下，ctxIdx 等于 276 的内容模型的初始化值表示为 pStateIdx = 63 和 valMPS = 0，这里 pStateIdx = 63 表示一种非自适应的概率状态。

Table 9-12 – Values of variables m and n for ctxIdx from 0 to 10

Initialisation variables	**ctxIdx**										
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
m	20	2	3	20	2	3	-28	-23	-6	-1	7
n	-15	54	74	-15	54	74	127	104	53	54	51

Table 9-13 – Values of variables m and n for ctxIdx from 11 to 23

Value of cabac_init_idc	Initialisation variables	ctxIdx												
		11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**
0	**m**	23	23	21	1	0	-37	5	-13	-11	1	12	-4	17
	n	33	2	0	9	49	118	57	78	65	62	49	73	50
1	**m**	22	34	16	-2	4	-29	2	-6	-13	5	9	-3	10
	n	25	0	0	9	41	118	65	71	79	52	50	70	54
2	**m**	29	25	14	-10	-3	-27	26	-4	-24	5	6	-17	14
	n	16	0	0	51	62	99	16	85	102	57	57	73	57

Table 9-14 – Values of variables m and n for ctxIdx from 24 to 39

Value of cabac_init_idc	Initialisation variables	ctxIdx															
		24	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
0	**m**	18	9	29	26	16	9	-46	-20	1	-13	-11	1	-6	-17	-6	9
	n	64	43	0	67	90	104	127	104	67	78	65	62	86	95	61	45
1	**m**	26	19	40	57	41	26	-45	-15	-4	-6	-13	5	6	-13	0	8
	n	34	22	0	2	36	69	127	101	76	71	79	52	69	90	52	43
2	**m**	20	20	29	54	37	12	-32	-22	-2	-4	-24	5	-6	-14	-6	4
	n	40	10	0	0	42	97	127	117	74	85	102	57	93	88	44	55

Table 9-15 – Values of variables m and n for ctxIdx from 40 to 53

Value of cabac_init_idc	Initialisation variables	ctxIdx													
		40	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**
0	**m**	-3	-6	-11	6	7	-5	2	0	-3	-10	5	4	-3	0
	n	69	81	96	55	67	86	88	58	76	94	54	69	81	88
1	**m**	-2	-5	-10	2	2	-3	-3	1	-3	-6	0	-3	-7	-5
	n	69	82	96	59	75	87	100	56	74	85	59	81	86	95
2	**m**	-11	-15	-21	19	20	4	6	1	-5	-13	5	6	-3	-1
	n	89	103	116	57	58	84	96	63	85	106	63	75	90	101

Table 9-16 – Values of variables m and n for ctxIdx from 54 to 59

Value of cabac_init_idc	Initialisation variables	ctxIdx					
		54	55	56	57	58	59
0	m	-7	-5	-4	-5	-7	1
	n	67	74	74	80	72	58
1	m	-1	-1	1	-2	-5	0
	n	66	77	70	86	72	61
2	m	3	-4	-2	-12	-7	1
	n	55	79	75	97	50	60

Table 9-17 – Values of variables m and n for ctxIdx from 60 to 69

Initialisation variables	ctxIdx									
	60	61	62	63	64	65	66	67	68	69
m	0	0	0	0	-9	4	0	-7	13	3
n	41	63	63	63	83	86	97	72	41	62

Table 9-18 – Values of variables m and n for ctxIdx from 70 to 104

ctxIdx	I and SI slices		Value of cabac_init_idc						ctxIdx	I and SI slices		Value of cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
70	0	11	0	45	13	15	7	34	88	-11	115	-13	108	-4	92	5	78
71	1	55	-4	78	7	51	-9	88	89	-12	63	-3	46	0	39	-6	55
72	0	69	-3	96	2	80	-20	127	90	-2	68	-1	65	0	65	4	61
73	-17	127	-27	126	-39	127	-36	127	91	-15	84	-1	57	-15	84	-14	83
74	-13	102	-28	98	-18	91	-17	91	92	-13	104	-9	93	-35	127	-37	127
75	0	82	-25	101	-17	96	-14	95	93	-3	70	-3	74	-2	73	-5	79
76	-7	74	-23	67	-26	81	-25	84	94	-8	93	-9	92	-12	104	-11	104
77	-21	107	-28	82	-35	98	-25	86	95	-10	90	-8	87	-9	91	-11	91
78	-27	127	-20	94	-24	102	-12	89	96	-30	127	-23	126	-31	127	-30	127
79	-31	127	-16	83	-23	97	-17	91	97	-1	74	5	54	3	55	0	65
80	-24	127	-22	110	-27	119	-31	127	98	-6	97	6	60	7	56	-2	79
81	-18	95	-21	91	-24	99	-14	76	99	-7	91	6	59	7	55	0	72
82	-27	127	-18	102	-21	110	-18	103	100	-20	127	6	69	8	61	-4	92
83	-21	114	-13	93	-18	102	-13	90	101	-4	56	-1	48	-3	53	-6	56
84	-30	127	-29	127	-36	127	-37	127	102	-5	82	0	68	0	68	3	68

85	-17	123	-7	92	0	80	11	80	**103**	-7	76	-4	69	-7	74	-8	71
86	-12	115	-5	89	-5	89	5	76	**104**	-22	125	-8	88	-9	88	-13	98
87	-16	122	-7	96	-7	94	2	84									

Table 9-19 – Values of variables m and n for ctxIdx from 105 to 165

ctxIdx	**I and SI slices**		**Value of cabac_init_idc**						**ctxIdx**	**I and SI slices**		**Value of cabac_init_idc**					
			0		**1**		**2**					**0**		**1**		**2**	
	m	**n**	**m**	**n**	**m**	**n**	**m**	**n**		**m**	**n**	**m**	**n**	**m**	**n**	**m**	**n**
105	-7	93	-2	85	-13	103	-4	86	**136**	-13	101	5	53	0	58	-5	75
106	-11	87	-6	78	-13	91	-12	88	**137**	-13	91	-2	61	-1	60	-8	80
107	-3	77	-1	75	-9	89	-5	82	**138**	-12	94	0	56	-3	61	-21	83
108	-5	71	-7	77	-14	92	-3	72	**139**	-10	88	0	56	-8	67	-21	64
109	-4	63	2	54	-8	76	-4	67	**140**	-16	84	-13	63	-25	84	-13	31
110	-4	68	5	50	-12	87	-8	72	**141**	-10	86	-5	60	-14	74	-25	64
111	-12	84	-3	68	-23	110	-16	89	**142**	-7	83	-1	62	-5	65	-29	94
112	-7	62	1	50	-24	105	-9	69	**143**	-13	87	4	57	5	52	9	75
113	-7	65	6	42	-10	78	-1	59	**144**	-19	94	-6	69	2	57	17	63
114	8	61	-4	81	-20	112	5	66	**145**	1	70	4	57	0	61	-8	74
115	5	56	1	63	-17	99	4	57	**146**	0	72	14	39	-9	69	-5	35
116	-2	66	-4	70	-78	127	-4	71	**147**	-5	74	4	51	-11	70	-2	27
117	1	64	0	67	-70	127	-2	71	**148**	18	59	13	68	18	55	13	91
118	0	61	2	57	-50	127	2	58	**149**	-8	102	3	64	-4	71	3	65
119	-2	78	-2	76	-46	127	-1	74	**150**	-15	100	1	61	0	58	-7	69
120	1	50	11	35	-4	66	-4	44	**151**	0	95	9	63	7	61	8	77
121	7	52	4	64	-5	78	-1	69	**152**	-4	75	7	50	9	41	-10	66
122	10	35	1	61	-4	71	0	62	**153**	2	72	16	39	18	25	3	62
123	0	44	11	35	-8	72	-7	51	**154**	-11	75	5	44	9	32	-3	68
124	11	38	18	25	2	59	-4	47	**155**	-3	71	4	52	5	43	-20	81
125	1	45	12	24	-1	55	-6	42	**156**	15	46	11	48	9	47	0	30
126	0	46	13	29	-7	70	-3	41	**157**	-13	69	-5	60	0	44	1	7
127	5	44	13	36	-6	75	-6	53	**158**	0	62	-1	59	0	51	-3	23
128	31	17	-10	93	-8	89	8	76	**159**	0	65	0	59	2	46	-21	74
129	1	51	-7	73	-34	119	-9	78	**160**	21	37	22	33	19	38	16	66
130	7	50	-2	73	-3	75	-11	83	**161**	-15	72	5	44	-4	66	-23	124
131	28	19	13	46	32	20	9	52	**162**	9	57	14	43	15	38	17	37
132	16	33	9	49	30	22	0	67	**163**	16	54	-1	78	12	42	44	-18

133	14	62	-7	100	-44	127	-5	90	**164**	0	62	0	60	9	34	50	-34
134	-13	108	9	53	0	54	1	67	**165**	12	72	9	69	0	89	-22	127
135	-15	100	2	53	-5	61	-15	72									

Table 9-20 – Values of variables m and n for ctxIdx from 166 to 226

ctxIdx	**I and SI slices**		**Value of cabac_init_idc**						**ctxIdx**	**I and SI slices**		**Value of cabac_init_idc**					
			0		**1**		**2**					**0**		**1**		**2**	
	m	**n**	**m**	**n**	**m**	**n**	**m**	**n**		**m**	**n**	**m**	**n**	**m**	**n**	**m**	**n**
166	24	0	11	28	4	45	4	39	**197**	26	-17	28	3	36	-28	28	-3
167	15	9	2	40	10	28	0	42	**198**	30	-25	28	4	38	-28	24	10
168	8	25	3	44	10	31	7	34	**199**	28	-20	32	0	38	-27	27	0
169	13	18	0	49	33	-11	11	29	**200**	33	-23	34	-1	34	-18	34	-14
170	15	9	0	46	52	-43	8	31	**201**	37	-27	30	6	35	-16	52	-44
171	13	19	2	44	18	15	6	37	**202**	33	-23	30	6	34	-14	39	-24
172	10	37	2	51	28	0	7	42	**203**	40	-28	32	9	32	-8	19	17
173	12	18	0	47	35	-22	3	40	**204**	38	-17	31	19	37	-6	31	25
174	6	29	4	39	38	-25	8	33	**205**	33	-11	26	27	35	0	36	29
175	20	33	2	62	34	0	13	43	**206**	40	-15	26	30	30	10	24	33
176	15	30	6	46	39	-18	13	36	**207**	41	-6	37	20	28	18	34	15
177	4	45	0	54	32	-12	4	47	**208**	38	1	28	34	26	25	30	20
178	1	58	3	54	102	-94	3	55	**209**	41	17	17	70	29	41	22	73
179	0	62	2	58	0	0	2	58	**210**	30	-6	1	67	0	75	20	34
180	7	61	4	63	56	-15	6	60	**211**	27	3	5	59	2	72	19	31
181	12	38	6	51	33	-4	8	44	**212**	26	22	9	67	8	77	27	44
182	11	45	6	57	29	10	11	44	**213**	37	-16	16	30	14	35	19	16
183	15	39	7	53	37	-5	14	42	**214**	35	-4	18	32	18	31	15	36
184	11	42	6	52	51	-29	7	48	**215**	38	-8	18	35	17	35	15	36
185	13	44	6	55	39	-9	4	56	**216**	38	-3	22	29	21	30	21	28
186	16	45	11	45	52	-34	4	52	**217**	37	3	24	31	17	45	25	21
187	12	41	14	36	69	-58	13	37	**218**	38	5	23	38	20	42	30	20
188	10	49	8	53	67	-63	9	49	**219**	42	0	18	43	18	45	31	12
189	30	34	-1	82	44	-5	19	58	**220**	35	16	20	41	27	26	27	16
190	18	42	7	55	32	7	10	48	**221**	39	22	11	63	16	54	24	42
191	10	55	-3	78	55	-29	12	45	**222**	14	48	9	59	7	66	0	93
192	17	51	15	46	32	1	0	69	**223**	27	37	9	64	16	56	14	56
193	17	46	22	31	0	0	20	33	**224**	21	60	-1	94	11	73	15	57

194	0	89	-1	84	27	36	8	63	**225**	12	68	-2	89	10	67	26	38
195	26	-19	25	7	33	-25	35	-18	**226**	2	97	-9	108	-10	116	-24	127
196	22	-17	30	-7	34	-30	33	-25									

Table 9-21 – Values of variables m and n for ctxIdx from 227 to 275

ctxIdx	**I and SI slices**		**Value of cabac_init_idc**						**ctxIdx**	**I and SI slices**		**Value of cabac_init_idc**					
			0		**1**		**2**					**0**		**1**		**2**	
	m	**n**	**m**	**n**	**m**	**n**	**m**	**n**		**m**	**n**	**m**	**n**	**m**	**n**	**m**	**n**
227	-3	71	-6	76	-23	112	-24	115	**252**	-12	73	-6	55	-16	72	-14	75
228	-6	42	-2	44	-15	71	-22	82	**253**	-8	76	0	58	-7	69	-10	79
229	-5	50	0	45	-7	61	-9	62	**254**	-7	80	0	64	-4	69	-9	83
230	-3	54	0	52	0	53	0	53	**255**	-9	88	-3	74	-5	74	-12	92
231	-2	62	-3	64	-5	66	0	59	**256**	-17	110	-10	90	-9	86	-18	108
232	0	58	-2	59	-11	77	-14	85	**257**	-11	97	0	70	2	66	-4	79
233	1	63	-4	70	-9	80	-13	89	**258**	-20	84	-4	29	-9	34	-22	69
234	-2	72	-4	75	-9	84	-13	94	**259**	-11	79	5	31	1	32	-16	75
235	-1	74	-8	82	-10	87	-11	92	**260**	-6	73	7	42	11	31	-2	58
236	-9	91	-17	102	-34	127	-29	127	**261**	-4	74	1	59	5	52	1	58
237	-5	67	-9	77	-21	101	-21	100	**262**	-13	86	-2	58	-2	55	-13	78
238	-5	27	3	24	-3	39	-14	57	**263**	-13	96	-3	72	-2	67	-9	83
239	-3	39	0	42	-5	53	-12	67	**264**	-11	97	-3	81	0	73	-4	81
240	-2	44	0	48	-7	61	-11	71	**265**	-19	117	-11	97	-8	89	-13	99
241	0	46	0	55	-11	75	-10	77	**266**	-8	78	0	58	3	52	-13	81
242	-16	64	-6	59	-15	77	-21	85	**267**	-5	33	8	5	7	4	-6	38
243	-8	68	-7	71	-17	91	-16	88	**268**	-4	48	10	14	10	8	-13	62
244	-10	78	-12	83	-25	107	-23	104	**269**	-2	53	14	18	17	8	-6	58
245	-6	77	-11	87	-25	111	-15	98	**270**	-3	62	13	27	16	19	-2	59
246	-10	86	-30	119	-28	122	-37	127	**271**	-13	71	2	40	3	37	-16	73
247	-12	92	1	58	-11	76	-10	82	**272**	-10	79	0	58	-1	61	-10	76
248	-15	55	-3	29	-10	44	-8	48	**273**	-12	86	-3	70	-5	73	-13	86
249	-10	60	-1	36	-10	52	-8	61	**274**	-13	90	-6	79	-1	70	-9	83
250	-6	62	1	38	-10	57	-8	66	**275**	-14	97	-8	85	-4	78	-10	87
251	-4	65	2	43	-9	58	-7	70									

Table 9-22 – Values of variables m and n for ctxIdx from 277 to 337

ctxIdx	I and SI slices		Value of cabac_init_idc						ctxIdx	I and SI slices		Value of cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
277	-6	93	-13	106	-21	126	-22	127	**308**	-16	96	-1	51	-16	77	-10	67
278	-6	84	-16	106	-23	124	-25	127	**309**	-7	88	7	49	-2	64	1	68
279	-8	79	-10	87	-20	110	-25	120	**310**	-8	85	8	52	2	61	0	77
280	0	66	-21	114	-26	126	-27	127	**311**	-7	85	9	41	-6	67	2	64
281	-1	71	-18	110	-25	124	-19	114	**312**	-9	85	6	47	-3	64	0	68
282	0	62	-14	98	-17	105	-23	117	**313**	-13	88	2	55	2	57	-5	78
283	-2	60	-22	110	-27	121	-25	118	**314**	4	66	13	41	-3	65	7	55
284	-2	59	-21	106	-27	117	-26	117	**315**	-3	77	10	44	-3	66	5	59
285	-5	75	-18	103	-17	102	-24	113	**316**	-3	76	6	50	0	62	2	65
286	-3	62	-21	107	-26	117	-28	118	**317**	-6	76	5	53	9	51	14	54
287	-4	58	-23	108	-27	116	-31	120	**318**	10	58	13	49	-1	66	15	44
288	-9	66	-26	112	-33	122	-37	124	**319**	-1	76	4	63	-2	71	5	60
289	-1	79	-10	96	-10	95	-10	94	**320**	-1	83	6	64	-2	75	2	70
290	0	71	-12	95	-14	100	-15	102	**321**	-7	99	-2	69	-1	70	-2	76
291	3	68	-5	91	-8	95	-10	99	**322**	-14	95	-2	59	-9	72	-18	86
292	10	44	-9	93	-17	111	-13	106	**323**	2	95	6	70	14	60	12	70
293	-7	62	-22	94	-28	114	-50	127	**324**	0	76	10	44	16	37	5	64
294	15	36	-5	86	-6	89	-5	92	**325**	-5	74	9	31	0	47	-12	70
295	14	40	9	67	-2	80	17	57	**326**	0	70	12	43	18	35	11	55
296	16	27	-4	80	-4	82	-5	86	**327**	-11	75	3	53	11	37	5	56
297	12	29	-10	85	-9	85	-13	94	**328**	1	68	14	34	12	41	0	69
298	1	44	-1	70	-8	81	-12	91	**329**	0	65	10	38	10	41	2	65
299	20	36	7	60	-1	72	-2	77	**330**	-14	73	-3	52	2	48	-6	74
300	18	32	9	58	5	64	0	71	**331**	3	62	13	40	12	41	5	54
301	5	42	5	61	1	67	-1	73	**332**	4	62	17	32	13	41	7	54
302	1	48	12	50	9	56	4	64	**333**	-1	68	7	44	0	59	-6	76
303	10	62	15	50	0	69	-7	81	**334**	-13	75	7	38	3	50	-11	82
304	17	46	18	49	1	69	5	64	**335**	11	55	13	50	19	40	-2	77
305	9	64	17	54	7	69	15	57	**336**	5	64	10	57	3	66	-2	77
306	-12	104	10	41	-7	69	1	67	**337**	12	70	26	43	18	50	25	42
307	-11	97	7	46	-6	67	0	68									

Table 9-23 – Values of variables m and n for ctxIdx from 338 to 398

ctxIdx	I and SI slices		Value of cabac_init_idc						ctxIdx	I and SI slices		Value of cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
338	15	6	14	11	19	-6	17	-13	**369**	32	-26	31	-4	40	-37	37	-17
339	6	19	11	14	18	-6	16	-9	**370**	37	-30	27	6	38	-30	32	1
340	7	16	9	11	14	0	17	-12	**371**	44	-32	34	8	46	-33	34	15
341	12	14	18	11	26	-12	27	-21	**372**	34	-18	30	10	42	-30	29	15
342	18	13	21	9	31	-16	37	-30	**373**	34	-15	24	22	40	-24	24	25
343	13	11	23	-2	33	-25	41	-40	**374**	40	-15	33	19	49	-29	34	22
344	13	15	32	-15	33	-22	42	-41	**375**	33	-7	22	32	38	-12	31	16
345	15	16	32	-15	37	-28	48	-47	**376**	35	-5	26	31	40	-10	35	18
346	12	23	34	-21	39	-30	39	-32	**377**	33	0	21	41	38	-3	31	28
347	13	23	39	-23	42	-30	46	-40	**378**	38	2	26	44	46	-5	33	41
348	15	20	42	-33	47	-42	52	-51	**379**	33	13	23	47	31	20	36	28
349	14	26	41	-31	45	-36	46	-41	**380**	23	35	16	65	29	30	27	47
350	14	44	46	-28	49	-34	52	-39	**381**	13	58	14	71	25	44	21	62
351	17	40	38	-12	41	-17	43	-19	**382**	29	-3	8	60	12	48	18	31
352	17	47	21	29	32	9	32	11	**383**	26	0	6	63	11	49	19	26
353	24	17	45	-24	69	-71	61	-55	**384**	22	30	17	65	26	45	36	24
354	21	21	53	-45	63	-63	56	-46	**385**	31	-7	21	24	22	22	24	23
355	25	22	48	-26	66	-64	62	-50	**386**	35	-15	23	20	23	22	27	16
356	31	27	65	-43	77	-74	81	-67	**387**	34	-3	26	23	27	21	24	30
357	22	29	43	-19	54	-39	45	-20	**388**	34	3	27	32	33	20	31	29
358	19	35	39	-10	52	-35	35	-2	**389**	36	-1	28	23	26	28	22	41
359	14	50	30	9	41	-10	28	15	**390**	34	5	28	24	30	24	22	42
360	10	57	18	26	36	0	34	1	**391**	32	11	23	40	27	34	16	60
361	7	63	20	27	40	-1	39	1	**392**	35	5	24	32	18	42	15	52
362	-2	77	0	57	30	14	30	17	**393**	34	12	28	29	25	39	14	60
363	-4	82	-14	82	28	26	20	38	**394**	39	11	23	42	18	50	3	78
364	-3	94	-5	75	23	37	18	45	**395**	30	29	19	57	12	70	-16	123
365	9	69	-19	97	12	55	15	54	**396**	34	26	22	53	21	54	21	53
366	-12	109	-35	125	11	65	0	79	**397**	29	39	22	61	14	71	22	56
367	36	-35	27	0	37	-33	36	-16	**398**	19	66	11	86	11	83	25	61
368	36	-34	28	0	39	-36	37	-14									

9.3.1.2 Initialisation process for the arithmetic decoding engine

9.3.1.2 算术解码引擎的初始化过程

This process is invoked before decoding the first macroblock of a slice or after the decoding of the pcm_alignment_zero_bit and all pcm_byte data for a macroblock of type I_PCM.

在解码 slice 中的第一个宏块之前调用该过程，或者解码完 pcm_alignment_zero_bit 和 I_PCM 宏块中的所有 pcm_byte 数据之后调用该过程。

Outputs of this process are the initialised decoding engine registers codIRange and codIOffset both in 16 bit register precision.

本过程的输出是初始化后的解码器引擎记录 codIRange 和 codIOffset，它们都是 16 位精度。

The status of the arithmetic decoding engine is represented by the variables codIRange and codIOffset. In the initialisation procedure of the arithmetic decoding process, codIRange is set equal to 0x01FE and codIOffset is set equal to the value returned from read_bits(9) interpreted as a 9 bit binary representation of an unsigned integer with most significant bit written first.

算术解码器的状态由变量 codIRange 和 codIOffset 来表示，在算术解码初始化的过程中，codIRange 被设置成 0x01FE，codIOffset 被设置成 read_bits(9) 的返回值，它是 9 位无符号整数二进制的表示。

NOTE – The description of the arithmetic decoding engine in this Recommendation | International Standard utilizes 16 bit register precision. However, the minimum register precision for the variables codIRange and codIOffset is 9 bits.

注意：本协议中算术解码引擎的描述使用的是 16 位寄存器精度，但是对于变量 codIRange 和 codIOffset 最小寄存器精度要求是 9 位。

9.3.2 Binarization process

9.3.2 二值化过程

Input to this process is a request for a syntax element.

本过程的输入是解码语法元素的需求。

Output of this process is the binarization of the syntax element, maxBinIdxCtx, ctxIdxOffset, and bypassFlag.

本过程的输出是语法元素的二进制表示、maxBinIdxCtx，ctxIdxOffset 和 bypassFlag。

Table 9-24 specifies the type of binarization process, maxBinIdxCtx, and ctxIdxOffset associated with each syntax element.

表 Table 9-24 说明了与每一个语法元素相关的二值化过程的类型、maxBinIdxCtx 和 ctxIdxOffset。

The specification of the unary (U) binarization process, the truncated unary (TU) binarization process, the concatenated unary / k-th order Exp-Golomb (UEGk) binarization process, and the fixed-length (FL) binarization process are given in subclauses 9.3.2.1 to 9.3.2.4, respectively. Other binarizations are specified in subclauses 9.3.2.5 to 9.3.2.7.

一元二值化过程、截断的一元二值化过程和串连/ k 阶 Exp-Golomb (UEGk) 码的二值化过程和定长的二值化过程分别在9.3.2.1 到 9.3.2.4 小节中说明，其他的二值化过程在 9.3.2.5 到 9.3.2.7 小节中说明。

Except for I slices, the binarizations for the syntax element mb_type as specified in subclause 9.3.2.5 consist of bin strings given by a concatenation of prefix and suffix bit strings. The UEGk binarization as specified in 9.3.2.3, which is used for the binarization of the syntax elements mvd_lX (X = 0, 1) and coeff_abs_level_minus1, and the binarization of the coded_block_pattern also consist of a concatenation of prefix and suffix bit strings. For these binarization processes, the prefix and the suffix bit string are separately indexed using the binIdx variable as specified further in subclause 9.3.3. The two sets of prefix bit strings and suffix bit strings are referred to as the binarization prefix part and the binarization suffix part, respectively.

除了 I slice 之外，语法元素 mb_type 的二值化过程在 9.3.2.5 小节中说明，由前缀比特串和后缀比特串串连组成，UEGk 的二值化过程在 9.3.2.3 小节中说明，主要用于语法元素 mvd_lX (X = 0, 1) 、coeff_abs_level_minus1 和 coded_block_pattern 的二值化过程，同样也是由前缀比特串和后缀比特串串连组成。对于这些二值化过程，前缀比特串和后缀比特串使用变量 binIdx 分别进行索引，如 9.3.3 小节所示。前缀比特串和后缀比特串的两个集分别称为二进制前缀部分和二进制后缀部分。

Associated with each binarization or binarization part is a specific value of the context index offset (ctxIdxOffset) variable and a specific value of the maxBinIdxCtx variable as given in Table 9-24. When two values for each of these variables are specified for one syntax element in Table 9-24, the value in the upper row is related to the prefix part while the value in the lower row is related to the suffix part of the binarization of the corresponding syntax element.

一个语法元素的每一个二进制位或者二进制部分都和一个内容索引的偏移量 (ctxIdxOffset) 以及 maxBinIdxCtx 变量有关，如表 Table 9-24 所示，对于一个语法元素来说，如果在 Table 9-24 中同时出现两个 maxBinIdxCtx 和两个 ctxIdxOffset 值时，那么上面的值和对应语法元素的前缀部分的有关，下面部分的值和对应语法元素的后缀部分有关。

The use of the DecodeBypass process and the variable bypassFlag is derived as follows.

DecodeBypass 过程的使用和变量 bypassFlag 的计算如下：

- If no value is assigned to ctxIdxOffset for the corresponding binarization or binarization part in Table 9-24 labelled as “na”, all bins of the bit strings of the corresponding binarization or of the binarization prefix/suffix part shall be decoded by invoking the DecodeBypass process as specified in subclause 9.3.3.2.3. In such a case, bypassFlag is set equal to 1, where bypassFlag is used to indicate that for parsing the value of the bin from the bitstream the DecodeBypass process shall be applied.

- 如果在 Table 9-24 中没有 ctxIdxOffset 的值来和相应的二进制位或者二进制部分对应，那么比特串的所有比特位的解码将使用 DecodeBypass 的解码过程，如 9.3.3.2.3 小节所示，在这种情况下 bypassFlag 被设置成 1，这里 bypassFlag 表示在解析二进制位时使用 DecodeBypass 的解码过程。

- Otherwise, for each possible value of binIdx up to the specified value of MaxBinIdxCtx given in Table 9-24, a specific value of the variable ctxIdx is further specified in subclause 9.3.3. bypassFlag is set equal to 0.

- 否则，对于每一个可能值 binIdx （最大一直到 MaxBinIdxCtx，如表 Table 9-24 所示），具体的 ctxIdx 在 9.3.3 小节中做进一步的说明，bypassFlag 被设置成 0。

The possible values of the context index ctxIdx are in the range of 0 to 398, inclusive. The value assigned to ctxIdxOffset specifies the lower value of the range of ctxIdx assigned to the corresponding binarization or binarization part of a syntax element.

内容索引的可能范围为 0 到 398，分配给 ctxIdxOffset 的值表示了对应语法元素二值化部分的 ctxIdx 范围的下限。

ctxIdx = ctxIdxOffset = 276 is assigned to the syntax element end_of_slice_flag and the bin of mb_type, which specifies the I_PCM macroblock type as further specified in subclause 9.3.3.1. For parsing the value of the corresponding bin from the bitstream, the arithmetic decoding process for decisions before termination (DecodeTerminate) as specified in subclause 9.3.3.2.4 shall be applied.

ctxIdx = ctxIdxOffset = 276 分配给语法元素 end_of_slice_flag 和 mb_type（表示 I_PCM 宏块类型） 的二进制位，在 9.3.3.1 小节中将做进一步的说明，对于从码流中使用算术解码方法解码语法元素时，解码结束前(DecodeTerminate) 的算术解码过程的说明见 9.3.3.2.4 小节。

NOTE – The bins of mb_type in I slices and the bins of the suffix for mb_type in SI slices that correspond to the same value of binIdx share the same ctxIdx. The last bin of the prefix of mb_type and the first bin of the suffix of mb_type in P, SP, and B slices may share the same ctxIdx.

注意：在 I slice 中的 mb_type 二进制位和 SI slice 中的 mb_type 前缀部分具有相同 binIdx 二进制位使用相同的 ctxIdx 。在 P, SP, 和 B slices 中，mb_type 前缀部分的最后一个二进制位和后缀部分的第一个二进制位使用相同的 ctxIdx 。

Table 9-24 – Syntax elements and associated types of binarization, maxBinIdxCtx, and ctxIdxOffset

Syntax element	Type of binarization	maxBinIdxCtx	ctxIdxOffset
mb_type (SI slices only)	prefix and suffix as specified in subclause 9.3.2.5	prefix: 0 suffix: 6	prefix: 0 suffix: 3
mb_type (I slices only)	as specified in subclause 9.3.2.5	6	3
mb_skip_flag (P, SP slices only)	FL, cMax=1	0	11
mb_type (P, SP slices only)	prefix and suffix as specified in subclause 9.3.2.5	prefix: 2 suffix: 5	prefix: 14 suffix: 17
sub_mb_type (P, SP slices only)	as specified in subclause 9.3.2.5	2	21
mb_skip_flag (B slices only)	FL, cMax=1	0	24
mb_type (B slices only)	prefix and suffix as specified in subclause 9.3.2.5	prefix: 3 suffix: 5	prefix: 27 suffix: 32
sub_mb_type (B slices only)	as specified in subclause 9.3.2.5	3	36
mvd_l0[ ][ ][0], mvd_l1[ ][ ][0]	prefix and suffix as given by UEG3 with signedValFlag=1, uCoff=9	prefix: 4 suffix: na	prefix: 40 suffix: na (uses DecodeBypass)
mvd_l0[ ][ ][1], mvd_l1[ ][ ][1]		prefix: 4 suffix: na	prefix: 47 suffix: na (uses DecodeBypass)
ref_idx_l0, ref_idx_l1	U	2	54
mb_qp_delta	as specified in subclause 9.3.2.7	2	60
intra_chroma_pred_mode	TU, cMax=3	1	64
prev_intra4x4_pred_mode_flag	FL, cMax=1	0	68
rem_intra4x4_pred_mode	FL, cMax=7	0	69
mb_field_decoding_flag	FL, cMax=1	0	70
coded_block_pattern	prefix and suffix as specified in subclause 9.3.2.6	prefix: 3 suffix: 1	prefix: 73 suffix: 77
coded_block_flag	FL, cMax=1	0	85
significant_coeff_flag (frame coded blocks only)	FL, cMax=1	0	105
last_significant_coeff_flag (frame coded blocks only)	FL, cMax=1	0	166
coeff_abs_level_minus1	prefix and suffix as given by UEG0 with signedValFlag=0, uCoff=14	prefix: 1 suffix: na	prefix: 227 suffix: na, (uses DecodeBypass)
coeff_sign_flag	FL, cMax=1	0	na, (uses DecodeBypass)
end_of_slice_flag	FL, cMax=1	0	276
significant_coeff_flag (field coded blocks only)	FL, cMax=1	0	277
last_significant_coeff_flag (field coded blocks only)	FL, cMax=1	0	338

9.3.2.1 Unary (U) binarization process

9.3.2.1 一元（U）二值化过程：

Input to this process is a request for a U binarization for a syntax element.

本过程的输入是：语法元素的一元二值化需求。

Output of this process is the U binarization of the syntax element.

本过程的输出是语法元素的二值化后的二进制位。

The bin string of a syntax element having (unsigned integer) value synElVal is a bit string of length synElVal + 1 indexed by BinIdx. The bins for binIdx less than synElVal are equal to 1. The bin with binIdx equal to synElVal is equal to 0.

值为 synElVal 的语法元素的二进制串是长度为 synElVal + 1 的二进制串，索引为 BinIdx，binIdx 小于 synElVal 的二进制位等于 1，binIdx 等于 synElVal 的二进制位等于 0。

Table 9-25 illustrates the bin strings of the unary binarization for a syntax element.

Table 9-25 说明了语法元素一元二值化后的二进制串。

Table 9-25 – Bin string of the unary binarization (informative)

Value of syntax element	Bin string					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
…						
binIdx	0	1	2	3	4	5

9.3.2.2 Truncated unary (TU) binarization process

9.3.2.2 截断一元（TU）二值化过程

Input to this process is a request for a TU binarization for a syntax element and cMax.

本过程的输入是语法元素的截断一元（TU）二值化需求和 cMax 。

Output of this process is the TU binarization of the syntax element.

本过程的输出是语法元素的截断一元二值化后的二进制比特位。

For syntax element (unsigned integer) values less than cMax, the U binarization process as specified in subclause 9.3.2.1 is invoked. For the syntax element value equal to cMax the bin string is a bit string of length cMax with all bins being equal to 1.

对于小于 cMax 的语法元素，仍然使用 9.3.2.1 小节中的二值化过程，对于等于 cMax 的语法元素，二进制比特串由长度为 cMax 值为 1 的二进制位组成。

NOTE – TU binarization is always invoked with a cMax value equal to the largest possible value of the syntax element being decoded.

注意：在 TU 二值化的过程中，cMax 表示解码语法元素的最大可能值。

9.3.2.3 Concatenated unary/ k-th order Exp-Golomb (UEGk) binarization process

9.3.2.3 串连一元/ K 阶 Exp-Golomb (UEGk) 的二值化过程。

Input to this process is a request for a UEGk binarization for a syntax element, signedValFlag and uCoff.

本过程的输入是需要二值化的语法元素、signedValFlag 和 uCoff。

Output of this process is the UEGk binarization of the syntax element.

本过程的输出是语法元素二值化后的二进制位。

A UEGk bin string is a concatenation of a prefix bit string and a suffix bit string. The prefix of the binarization is specified by invoking the TU binarization process for the prefix part Min(uCoff, Abs(synElVal)) of a syntax element value synElVal as specified in subclause 9.3.2.2 with cMax = uCoff, where uCoff > 0.

一个 UEGk 比特串是由前缀比特串和后缀比特串组成，前缀比特串由 TU 二值化过程得到，将 Min(uCoff, Abs(synElVal)) 和 cMax = uCoff 作为输入，这里 uCoff > 0，synElVal 表示语法元素的值。

The UEGk bin string is derived as follows.

UEGk 二进制串的计算如下：

- If one of the following is true, the bin string of a syntax element having value synElVal consists only of a prefix bit string,
- 如果下面的一个条件满足，那么值为 synElVal 语法元素的二进制串仅由前缀比特串组成
  - signedValFlag is equal to 0 and the prefix bit string is not equal to the bit string of length uCoff with all bits equal to 1.
  - signedValFlag 等于 0，前缀比特串不等于长度为 uCoff 值全部为 1 的二进制串。
  - signedValFlag is equal to 1 and the prefix bit string is equal to the bit string that consists of a single bit with value equal to 0.
  - signedValFlag 等于 1，前缀比特串等于
- Otherwise, the bin string of the UEGk suffix part of a syntax element value synElVal is specified by a process equivalent to the following pseudo-code:
- 否则，值为 synElVal 的语法元素后缀部分的 UEGk 二进制串的计算如下面的伪代码所示。

```
if( Abs( synElVal ) >= uCoff ) {
    sufS = Abs( synElVal ) − uCoff
    stopLoop = 0
    do {
        if( sufS >= ( 1 << k ) ) {
            put( 1 )
            sufS = sufS − ( 1<<k )
            k++
        } else {
            put( 0 )
            while( k− − )
                put( ( sufS >> k ) & 0x01 )
            stopLoop = 1
        }
    } while( !stopLoop )
}
if( signedValFlag && synElVal != 0)
    if( synElVal > 0 )
        put( 0 )
    else
        put( 1 )
```

NOTE – The specification for the k-th order Exp-Golomb (EGk) code uses 1's and 0's in reverse meaning for the unary part of the Exp-Golomb code of 0-th order as specified in subclause 9.1.

注意：K 阶 Exp-Golomb (EGk) 码的说明使用 1 和 0 来表示，它们刚好是 9.1 小节中 0 阶 Exp-Golomb 码一元部分的反意。（需要结合伪代码解释）

9.3.2.4 Fixed-length (FL) binarization process

9.3.2.4 定长 (FL) 二值化过程

Input to this process is a request for a FL binarization for a syntax element and cMax.

本过程的输入是需要定长二值化的语法元素和 cMax。

Output of this process is the FL binarization of the syntax element.

本过程的输出是语法元素的定长二值化后的二进制位。

FL binarization is constructed by using an fixedLength-bit unsigned integer bin string of the syntax element value, where fixedLength = Ceil(Log2(cMax + 1)). The indexing of bins for the FL binarization is such that the binIdx = 0 relates to the least significant bit with increasing values of binIdx towards the most significant bit.

FL 的二值化过程就是对语法元素的值使用定长 fixedLength-bit 的无符号二进制串来表示，这里 fixedLength = Ceil(Log2(cMax + 1))，在 FL 二值化过程中，二进制位的索引是这样的，最不重要位的索引为 binIdx = 0，然后随着重要性的增加，索引值也增加。

9.3.2.5 Binarization process for macroblock type and sub-macroblock type

9.3.2.5 宏块类型和子宏块类型的二值化过程

Input to this process is a request for a binarization for syntax elements mb_type or sub_mb_type.

本过程的输入是语法元素 mb_type 或者 sub_mb_type 二值化需求。

Output of this process is the binarization of the syntax element.

本过程的输出是语法元素二值化后的二进制位。

The binarization scheme for decoding of macroblock type in I slices is specified in Table 9-26.

在 I slice 中宏块类型的二值化方法见 Table 9-26 中的说明。

For macroblock types in SI slices, the binarization consists of bin strings specified as a concatenation of a prefix and a suffix bit string as follows.

对于 SI slice 中的宏块类型，二值化由前缀比特串和后缀比特串串连组成。

The prefix bit string consists of a single bit, which is specified by $b_0 = ((\text{mb\_type} == \text{SI}) ?\ 0 : 1)$. For the syntax element value for which b_0 is equal to 0, the bin string only consists of the prefix bit string. For the syntax element value for which b_0 is equal to 1, the binarization is given by concatenating the prefix b_0 and the suffix bit string as specified in Table 9-26 for macroblock type in I slices indexed by subtracting 1 from the value of mb_type in SI slices.

前缀比特串由一个比特组成，说明如下：$b_0 = ((\text{mb\_type} == \text{SI}) ?\ 0 : 1)$，对于 b_0 等于 0 的语法元素值来说，比特串只有前缀比特串组成；对于 b_0 等于 1 的语法元素值来说，比特串由前缀码 b_0 和后缀比特串组成，后缀比特串如 Table 9-26 所示，在 SI 中的索引需要减 1。

（在 SI 宏块中，只使用 4x4 的帧内预测方式，所以没有必要编码它的编码模式）

Table 9-26 – Binarization for macroblock types in I slices

Value (name) of mb_type	Bin string						
0 (I_4x4)	0						
1 (I_16x16_0_0_0)	1	0	0	0	0	0	
2 (I_16x16_1_0_0)	1	0	0	0	0	1	
3 (I_16x16_2_0_0)	1	0	0	0	1	0	
4 (I_16x16_3_0_0)	1	0	0	0	1	1	
5 (I_16x16_0_1_0)	1	0	0	1	0	0	0
6 (I_16x16_1_1_0)	1	0	0	1	0	0	1
7 (I_16x16_2_1_0)	1	0	0	1	0	1	0
8 (I_16x16_3_1_0)	1	0	0	1	0	1	1
9 (I_16x16_0_2_0)	1	0	0	1	1	0	0
10 (I_16x16_1_2_0)	1	0	0	1	1	0	1
11 (I_16x16_2_2_0)	1	0	0	1	1	1	0
12 (I_16x16_3_2_0)	1	0	0	1	1	1	1
13 (I_16x16_0_0_1)	1	0	1	0	0	0	
14 (I_16x16_1_0_1)	1	0	1	0	0	1	
15 (I_16x16_2_0_1)	1	0	1	0	1	0	
16 (I_16x16_3_0_1)	1	0	1	0	1	1	
17 (I_16x16_0_1_1)	1	0	1	1	0	0	0
18 (I_16x16_1_1_1)	1	0	1	1	0	0	1
19 (I_16x16_2_1_1)	1	0	1	1	0	1	0
20 (I_16x16_3_1_1)	1	0	1	1	0	1	1
21 (I_16x16_0_2_1)	1	0	1	1	1	0	0
22 (I_16x16_1_2_1)	1	0	1	1	1	0	1
23 (I_16x16_2_2_1)	1	0	1	1	1	1	0
24 (I_16x16_3_2_1)	1	0	1	1	1	1	1
25 (I_PCM)	1	1					
binIdx	0	1	2	3	4	5	6

The binarization schemes for P macroblock types in P and SP slices and for B macroblocks in B slices are specified in Table 9-27.

在 P 和 SP slice 中 P 宏块类型的二值化方法和 B slice 中的 B 宏块类型的二值化方法在 Table 9-27 中说明。

The bin string for I macroblock types in P and SP slices corresponding to mb_type values 5 to 30 consists of a concatenation of a prefix, which consists of a single bit with value equal to 1 as specified in Table 9-27 and a suffix as specified in Table 9-26, indexed by subtracting 5 from the value of mb_type.

在 P 和 SP slice 中 I 宏块的比特串对应于 mb_type 的值为 5 到 30，它也由一个前缀码和后缀码组成，等于 1 的前缀码在 Table 9-27 说明，后缀码如 Table 9-26 所示，索引为 mb_type 的值减 5。

mb_type equal to 4 (P_8x8ref0) is not allowed..

不允许使用等于 4 的 mb_type 。

For I macroblock types in B slices (mb_type values 23 to 48) the binarization consists of bin strings specified as a concatenation of a prefix bit string as specified in Table 9-27 and suffix bit strings as specified in Table 9-26, indexed by subtracting 23 from the value of mb_type.

对于 B slice 中的 I 宏块类型（mb_type 值的范围是 23 到 48），它也是由一个前缀码和一个后缀码组成，前缀码如表 Table 9-27，后缀码如表 Table 9-26，索引为 mb_type 减去 23。

Table 9-27 – Binarization for macroblock types in P, SP, and B slices

Slice type	Value (name) of mb_type	Bin string						
P, SP slice	0 (P_L0_16x16)	0	0	0				
	1 (P_L0_L0_16x8)	0	1	1				
	2 (P_L0_L0_8x16)	0	1	0				
	3 (P_8x8)	0	0	1				
	4 (P_8x8ref0)	na						
	5 to 30 (Intra, prefix only)	1						
B slice	0 (B_Direct_16x16)	0						
	1 (B_L0_16x16)	1	0	0				
	2 (B_L1_16x16)	1	0	1				
	3 (B_Bi_16x16)	1	1	0	0	0	0	
	4 (B_L0_L0_16x8)	1	1	0	0	0	1	
	5 (B_L0_L0_8x16)	1	1	0	0	1	0	
	6 (B_L1_L1_16x8)	1	1	0	0	1	1	
	7 (B_L1_L1_8x16)	1	1	0	1	0	0	
	8 (B_L0_L1_16x8)	1	1	0	1	0	1	
	9 (B_L0_L1_8x16)	1	1	0	1	1	0	
	10 (B_L1_L0_16x8)	1	1	0	1	1	1	
	11 (B_L1_L0_8x16)	1	1	1	1	1	0	
	12 (B_L0_Bi_16x8)	1	1	1	0	0	0	0
	13 (B_L0_Bi_8x16)	1	1	1	0	0	0	1
	14 (B_L1_Bi_16x8)	1	1	1	0	0	1	0
	15 (B_L1_Bi_8x16)	1	1	1	0	0	1	1
	16 (B_Bi_L0_16x8)	1	1	1	0	1	0	0
	17 (B_Bi_L0_8x16)	1	1	1	0	1	0	1
	18 (B_Bi_L1_16x8)	1	1	1	0	1	1	0
	19 (B_Bi_L1_8x16)	1	1	1	0	1	1	1
	20 (B_Bi_Bi_16x8)	1	1	1	1	0	0	0
	21 (B_Bi_Bi_8x16)	1	1	1	1	0	0	1
	22 (B_8x8)	1	1	1	1	1	1	
	23 to 48 (Intra, prefix only)	1	1	1	1	0	1	
binIdx		0	1	2	3	4	5	6

For P, SP, and B slices the specification of the binarization for sub_mb_type is given in Table 9-28.

对于 P, SP, 和 B slices 中的 sub_mb_type 的二值化由 Table 9-28 说明。

Table 9-28 – Binarization for sub-macroblock types in P, SP, and B slices

Slice type	Value (name) of sub_mb_type	Bin string					
P, SP slice	0 (P_L0_8x8)	1					
	1 (P_L0_8x4)	0	0				
	2 (P_L0_4x8)	0	1	1			
	3 (P_L0_4x4)	0	1	0			
B slice	0 (B_Direct_8x8)	0					
	1 (B_L0_8x8)	1	0	0			
	2 (B_L1_8x8)	1	0	1			
	3 (B_Bi_8x8)	1	1	0	0	0	
	4 (B_L0_8x4)	1	1	0	0	1	
	5 (B_L0_4x8)	1	1	0	1	0	
	6 (B_L1_8x4)	1	1	0	1	1	
	7 (B_L1_4x8)	1	1	1	0	0	0
	8 (B_Bi_8x4)	1	1	1	0	0	1
	9 (B_Bi_4x8)	1	1	1	0	1	0
	10 (B_L0_4x4)	1	1	1	0	1	1
	11 (B_L1_4x4)	1	1	1	1	0	
	12 (B_Bi_4x4)	1	1	1	1	1	
binIdx		0	1	2	3	4	5

9.3.2.6 Binarization process for coded block pattern

9.3.2.6 编码块模式的二值化过程

Input to this process is a request for a binarization for the syntax element coded_block_pattern.

本过程的输入是语法元素 coded_block_pattern 二值化的需求。

Output of this process is the binarization of the syntax element.

本过程的输出是语法元素二值化后的二进制。

The binarization of coded_block_pattern consists of a concatenation of a prefix part and a suffix part. The prefix part of the binarization is given by the FL binarization of CodedBlockPatternLuma with cMax = 15. The suffix part consists of the TU binarization of CodedBlockPatternChroma with cMax = 2. The relationship between the value of the syntax element coded_block_pattern and the values of CodedBlockPatternLuma and CodedBlockPatternChroma is given as specified in subclause 7.4.5.

coded_block_pattern 的二值化过程由一个前缀部分和一个后缀部分组成，前缀部分是定长二值化过程产生，它的输入参数为 CodedBlockPatternLuma 和 cMax = 15，后缀部分由截断二值化过程产生，输入参数为 CodedBlockPatternChroma 和 cMax = 2，语法元素 coded_block_pattern 与 CodedBlockPatternLuma 和 CodedBlockPatternChroma 的关系在 7.4.5 小节中说明。

9.3.2.7 Binarization process for mb_qp_delta

9.3.2.7 mb_qp_delta 的二值化处理过程

Input to this process is a request for a binarization for the syntax element mb_qp_delta.

本过程的输入是语法元素 mb_qp_delta 的二值化需求。

Output of this process is the binarization of the syntax element.

本过程的输出是语法元素二值化后的二进制位

The bin string of mb_qp_delta is derived by the U binarization of the mapped value of the syntax element mb_qp_delta, where the assignment rule between the signed value of mb_qp_delta and its mapped value is given as specified in Table 9-3.

mb_qp_delta 的二进制串由语法元素 mb_qp_delta 的映射值的 U 二值化计算得到，有符号的 mb_qp_delta 和它的映射值之间的分配规则由 Table 9-3 说明。

9.3.3 Decoding process flow

9.3.3 解码过程流程

Input to this process is a binarization of the requested syntax element, maxBinIdxCtx, bypassFlag and ctxIdxOffset as specified in subclause 9.3.2.

本过程的输出是需要解码的语法元素的二进制位、maxBinIdxCtx、bypassFlag 和 ctxIdxOffset，如 9.3.2 小节。

Output of this process is the value of the syntax element.

本过程的输出是语法元素的值。

This process specifies how each bit of a bit string is parsed for each syntax element.

本过程说明了对于每一个语法元素，比特串中的每一个比特位是如何解析的？

After parsing each bit, the resulting bit string is compared to all bin strings of the binarization of the syntax element and the following applies.

解析完每一个比特位之后，最后的比特串语法元素的二值化后的所有比特串进行比较，并且使用下面的过程。

- If the bit string is equal to one of the bin strings, the corresponding value of the syntax element is the output.
- 如果比特串等于一个二进制串，那么输出对应的语法元素的值。
- Otherwise (the bit string is not equal to one of the bin strings), the next bit is parsed.
- 否则（比特串不等于一个二进制串），那么继续解析下一个比特位。

While parsing each bin, the variable binIdx is incremented by 1 starting with binIdx being set equal to 0 for the first bin.

当解析每一个二进制位时，变量 binIdx 依次增加 1，开始解码时，binIdx 被设置成 0。

When the binarization of the corresponding syntax element consists of a prefix and a suffix binarization part,, the variable binIdx is set equal to 0 for the first bin of each part of the bin string (prefix part or suffix part). In this case, after parsing the prefix bit string, the parsing process of the suffix bit string related to the binarizations specified in subclauses 9.3.2.3 and 9.3.2.5 is invoked depending on the resulting prefix bit string as specified in subclauses 9.3.2.3 and 9.3.2.5. Note that for the binarization of the syntax element coded_block_pattern, the suffix bit string is present regardless of the prefix bit string of length 4 as specified in subclause 9.3.2.6.

当语法元素的二进制表示由前缀二进制部分和后缀二进制部分组成时，对于每一个二进制串的第一个二进制位来说（前缀部分或者后缀部分），变量 binIdx 都被设置成等于 0，在这种情况下，解析完前缀比特串之后，根据前缀比特串的解析结果，如 9.3.2.3 和 9.3.2.5 小节所示，再使用相关的后缀比特串的解析过程。注意，对于语法元素 coded_block_pattern 的二进制表示来说，后缀比特串的出现和前缀四比特位比特串无关，如 9.3.2.6 小节所示。

Depending on the variable bypassFlag, the following applies.

根据变量 bypassFlag 的值，使用下面的过程：

- If bypassFlag is equal to 1, the bypass decoding process as specified in subclause 9.3.3.2.3 shall be applied for parsing the value of the bins from the bitstream.
- 如果 bypassFlag 等于 1，那么使用 9.3.3.2.3 小节中的 bypass 解码过程，从码流中解析出比特位的值。
- Otherwise (bypassFlag is equal to 0), the parsing of each bin is specified by the following two ordered steps:
- 否则（bypassFlag 等于 0），每一个比特位的解析过程按照下面的两个步骤进行：

1. Given binIdx, maxBinIdxCtx and ctxIdxOffset, ctxIdx is derived as specified in subclause 9.3.3.1.

   根据给定的 binIdx，maxBinIdxCtx 和 ctxIdxOffset，使用 9.3.3.1 小节中的过程计算出 ctxIdx。

2. Given ctxIdx, the value of the bin from the bitstream as specified in subclause 9.3.3.2 is decoded.

   根据上面计算得到的 ctxIdx ，使用 9.3.3.2 小节中的过程从码流中解码出比特位的值。

9.3.3.1 Derivation process for ctxIdx

9.3.3.1 ctxIdx 的计算过程

Inputs to this process are binIdx, maxBinIdxCtx and ctxIdxOffset.

本过程的输出是 binIdx, maxBinIdxCtx 和 ctxIdxOffset。

Output of this process is ctxIdx.

本过程的输出是 ctxIdx。

Table 9-29 shows the assignment of ctxIdx increments (ctxIdxInc) to binIdx for all ctxIdxOffset values except those related to the syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1.

Table 9-29 表示了对于所有的 ctxIdxOffset 值来说 binIdx 的 ctxIdx 增量的分配过程， 除了语法元素 coded_block_flag，significant_coeff_flag，last_significant_coeff_flag 和 coeff_abs_level_minus1 之外。

The ctxIdx to be used with a specific binIdx is specified by first determining the ctxIdxOffset associated with the given bin string or part thereof. The ctxIdx is determined as follows.

对于与具体的 binIdx 一起使用的 ctxIdx 来说，首先计算出与二进制串或部分二进制串相关的 ctxIdxOffset，ctxIdx 的计算如下所示：

- If the ctxIdxOffset is listed in Table 9-29, the ctxIdx for a binIdx is the sum of ctxIdxOffset and ctxIdxInc, which is found in Table 9-29. When more than one value is listed in Table 9-29 for a binIdx, the assignment process for ctxIdxInc for that binIdx is further specified in the subclauses given in parenthesis of the corresponding table entry.
- 如果 ctxIdxOffset 在 Table 9-29 中已经列出，那么 binIdx 的 ctxIdx 是 ctxIdxOffset 和 ctxIdxInc 的和，如 Table 9-29 所示，如果在 Table 9-29 中，一个 binIdx 有多个值与它相对应，那么 ctxIdxInc 的分配过程将在圆括号中标出的小节中做进一步的描述。
- Otherwise (ctxIdxOffset is not listed in Table 9-29), the ctxIdx is specified to be the sum of the following terms: ctxIdxOffset and ctxIdxBlockCatOffset(ctxBlockCat) as specified in Table 9-30 and ctxIdxInc(ctxBlockCat). Subclause 9.3.3.1.3 specifies which ctxBlockCat is used. Subclause 9.3.3.1.1.9 specifies the assignment of ctxIdxInc(ctxBlockCat) for coded_block_flag, and subclause 9.3.3.1.3 specifies the assignment of ctxIdxInc(ctxBlockCat) for significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1.
- 否则（ctxIdxOffset 在 Table 9-29 中没有列出），那么 ctxIdx 是下面变量的和：ctxIdxOffset 和 ctxIdxBlockCatOffset(ctxBlockCat)，如 Table 9-30 所示，ctxIdxInc(ctxBlockCat)。9.3.3.1.3 小节说明了使用哪个 ctxBlockCat，9.3.3.1.1.9 小节说明了 coded_block_flag 语法元素的 ctxIdxInc(ctxBlockCat) 分配过程，9.3.3.1.3 小节说明了 significant_coeff_flag 、last_significant_coeff_flag 和 coeff_abs_level_minus1语法元素的 ctxIdxInc(ctxBlockCat) 分配过程。

All bins with binIdx larger than maxBinIdxCtx are parsed using ctxIdx assigned to maxBinIdxCtx.

所有大于 maxBinIdxCtx 的 binIdx 二进制位都使用分配给 maxBinIdxCtx 的 ctxIdx 来解析。

All entries in Table 9-29 labelled with “na” correspond to values of binIdx that do not occur for the corresponding ctxIdxOffset.

在 Table 9-29 中，所有标识为“na”的索引对应的 binIdx 都不可能发生。

ctxIdx = 276 is assigned to the binIdx of mb_type indicating the I_PCM mode. For parsing the value of the corresponding bins from the bitstream, the arithmetic decoding process for decisions before termination as specified in subclause 9.3.3.2.4 shall be applied.

ctxIdx = 276 被分配给 I_PCM 模式的 mb_type 的 binIdx。从码流中解析对应的二进制位的值时，终止解码之前的算术解码过程如 9.3.3.2.4 小节所示。

Table 9-29 – Assignment of ctxIdxInc to binIdx for all ctxIdxOffset values except those related to the syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1

ctxIdxOffset	binIdx						
	0	**1**	**2**	**3**	**4**	**5**	**>= 6**
0	0,1,2 (subclause 9.3.3.1.1.3)	na	na	na	na	na	na
3	0,1,2 (subclause 9.3.3.1.1.3)	ctxIdx=276	3	4	5,6 (subclause 9.3.3.1.2)	6,7 (subclause 9.3.3.1.2)	7
11	0,1,2 (subclause 9.3.3.1.1.1)	na	na	na	na	na	na
14	0	1	2,3 (subclause 9.3.3.1.2)	na	na	na	na
17	0	ctxIdx=276	1	2	2,3 (subclause 9.3.3.1.2)	3	3
21	0	1	2	na	na	na	na
24	0,1,2 (subclause 9.3.3.1.1.1)	na	na	na	na	na	na
27	0,1,2 (subclause 9.3.3.1.1.3)	3	4,5 (subclause 9.3.3.1.2)	5	5	5	5
32	0	ctxIdx=276	1	2	2,3 (subclause 9.3.3.1.2)	3	3
36	0	1	2,3 (subclause 9.3.3.1.2)	3	3	3	na
40	0,1,2 (subclause 9.3.3.1.1.7)	3	4	5	6	6	6
47	0,1,2 (subclause 9.3.3.1.1.7)	3	4	5	6	6	6
54	0,1,2,3 (subclause 9.3.3.1.1.6)	4	5	5	5	5	5
60	0,1 (subclause 9.3.3.1.1.5)	2	3	3	3	3	3
64	0,1,2 (subclause 9.3.3.1.1.8)	3	3	na	na	na	na
68	0	na	na	na	na	na	na
69	0	0	0	na	na	na	na
70	0,1,2 (subclause 9.3.3.1.1.2)	na	na	na	na	na	na
73	0,1,2,3 (subclause 9.3.3.1.1.4)	0,1,2,3 (subclause 9.3.3.1.1.4)	0,1,2,3 (subclause 9.3.3.1.1.4)	0,1,2,3 (subclause 9.3.3.1.1.4)	na	na	na
77	0,1,2,3 (subclause 9.3.3.1.1.4)	4,5,6,7 (subclause 9.3.3.1.1.4)	na	na	na	na	na
276	0	na	na	na	na	na	na

Table 9-30 shows the values of ctxIdxBlockCatOffset depending on ctxBlockCat for the syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1. The specification of ctxBlockCat is given in Table 9-32.

Table 9-30 表示了对于语法元素 coded_block_flag，significant_coeff_flag， last_significant_coeff_flag， 和 coeff_abs_level_minus1来说， 由 ctxBlockCat 计算 ctxIdxBlockCatOffset 值的过程， ctxBlockCat 的说明见表 Table 9-32。

Table 9-30 – Assignment of ctxIdxBlockCatOffset to ctxBlockCat for syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1

Syntax element	ctxBlockCat (as specified in Table 9-32)				
	0	**1**	**2**	**3**	**4**
coded_block_flag	0	4	8	12	16
significant_coeff_flag	0	15	29	44	47
last_significant_coeff_flag	0	15	29	44	47
coeff_abs_level_minus1	0	10	20	30	39

9.3.3.1.1 Assignment process of ctxIdxInc using neighbouring syntax elements

9.3.3.1.1 使用相邻的语法元素对 ctxIdxInc 的分配过程

Subclause 9.3.3.1.1.1 specifies the derivation process of ctxIdxInc for the syntax element mb_skip_flag.

9.3.3.1.1.1 小节说明了语法元素 mb_skip_flag 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.2 specifies the derivation process of ctxIdxInc for the syntax element mb_field_decoding_flag.

9.3.3.1.1.2 小节说明了语法元素 mb_field_decoding_flag 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.3 specifies the derivation process of ctxIdxInc for the syntax element mb_type.

9.3.3.1.1.3 小节说明了语法元素 mb_type 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.4 specifies the derivation process of ctxIdxInc for the syntax element coded_block_pattern.

9.3.3.1.1.4 小节说明了语法元素 coded_block_pattern 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.5 specifies the derivation process of ctxIdxInc for the syntax element mb_qp_delta.

9.3.3.1.1.5 小节说明了语法元素 mb_qp_delta 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.6 specifies the derivation process of ctxIdxInc for the syntax elements ref_idx_l0 and ref_idx_l1.

9.3.3.1.1.6 小节说明了语法元素 ref_idx_l0 和 ref_idx_l1 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.7 specifies the derivation process of ctxIdxInc for the syntax elements mvd_l0 and mvd_l1.

9.3.3.1.1.7 小节说明了语法元素 mvd_l0 和 mvd_l1 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.8 specifies the derivation process of ctxIdxInc for the syntax element intra_chroma_pred_mode.

9.3.3.1.1.8 小节说明了语法元素 intra_chroma_pred_mode 的 ctxIdxInc 计算方法。

Subclause 9.3.3.1.1.9 specifies the derivation process of ctxIdxInc for the syntax element coded_block_flag.

9.3.3.1.1.9 小节说明了语法元素 coded_block_flag 的 ctxIdxInc 计算方法。

9.3.3.1.1.1 Derivation process of ctxIdxInc for the syntax element mb_skip_flag

9.3.3.1.1.1 语法元素 mb_skip_flag 的 ctxIdxInc 的计算方法

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc 。

When MbaffFrameFlag is equal to 1 and mb_field_decoding_flag has not been decoded (yet) for the current macroblock pair with top macroblock address 2 * (CurrMbAddr / 2), the inference rule for the syntax element mb_field_decoding_flag as specified in subclause 7.4.4 shall be applied.

当 MbaffFrameFlag 等于 1 并且顶场宏块地址为 2 * (CurrMbAddr / 2) 的宏块对的 mb_field_decoding_flag 语法元素还没有解码，那么 mb_field_decoding_flag 的计算规则如 7.4.4 小节中的所示。

The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrA and mbAddrB.

使用 6.4.7.1 小节中的相邻宏块的计算过程，输出赋给 mbAddrA 和 mbAddrB。

Let the variable condTermFlagN (with N being either A or B) be derived as follows.

变量 condTermFlagN （N 可能是 A 或者 B）的计算如下：

- If mbAddrN is not available or mb_skip_flag for the macroblock mbAddrN is equal to 1, condTermFlagN is set equal to 0.
- 如果 mbAddrN 不可用或者宏块 mbAddrN 的 mb_skip_flag 等于 1，那么 condTermFlagN 设置成等于 0；
- Otherwise (mbAddrN is available and mb_skip_flag for the macroblock mbAddrN is equal to 0), condTermFlagN is set equal to 1.
- 否则（mbAddrN 可用并且宏块 mbAddrN 的 mb_skip_flag 等于 0），那么 condTermFlagN 设置成等于 1；

The variable ctxIdxInc is derived by

变量 ctxIdxInc 的计算如下：

$$ctxIdxInc = condTermFlagA + condTermFlagB \qquad (9\text{-}1)$$

9.3.3.1.1.2 Derivation process of ctxIdxInc for the syntax element mb_field_decoding_flag

9.3.3.1.1.2 语法元素 mb_field_decoding_flag 的 ctxIdxInc 的计算方法

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

The derivation process for neighbouring macroblock addresses and their availability in MBAFF frames as specified in subclause 6.4.6 is invoked and the output is assigned to mbAddrA and mbAddrB.

使用 6.4.6 小节中说明的 MBAFF 帧中相邻宏块地址和它们可用性的计算过程，输出赋给 mbAddrA 和 mbAddrB。

Let the variable condTermFlagN (with N being either A or B) be derived as follows.

变量 condTermFlagN （N 可能是 A 或者 B）的计算方法如下：

- When both macroblocks mbAddrN and mbAddrN + 1 have mb_type equal to P_Skip or B_Skip, the inference rule for the syntax element mb_field_decoding_flag as specified in subclause 7.4.4 shall be applied for the macroblock mbAddrN.
- 当宏块 mbAddrN 和 mbAddrN + 1 的 mb_type 为 P_Skip 或者 B_Skip，那么使用 7.4.4 小节中的方法计算出宏块 mbAddrN 中语法元素 mb_field_decoding_flag 的值；
- If any of the following conditions is true, condTermFlagN is set equal to 0,
- 如果下面的任何一个条件成立，那么 condTermFlagN 被设置成 0；
    - mbAddrN is not available

      mbAddrN 不可用；
    - the macroblock mbAddrN is a frame macroblock.
    - 宏块 mbAddrN 是帧结构宏块；
- Otherwise, condTermFlagN is set equal to 1.
- 否则，condTermFlagN 设置成等于 1。

The variable ctxIdxInc is derived by

变量 ctxIdxInc 的计算如下：

ctxIdxInc = condTermFlagA + condTermFlagB (9-2)

9.3.3.1.1.3 Derivation process of ctxIdxInc for the syntax element mb_type

语法元素 mb_type 的 ctxIdxInc 的计算方法

Input to this process is ctxIdxOffset.

本过程的输入是 ctxIdxOffset。

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrA and mbAddrB.

使用 6.4.7.1 小节中的方法计算出相邻的宏块并且将输出赋给 mbAddrA 和 mbAddrB。

Let the variable condTermFlagN (with N being either A or B) be derived as follows.

变量 condTermFlagN (N 可能为 A 或 B) 的计算过程如下：

- If any of the following conditions is true, condTermFlagN is set equal to 0
- 如果下面任何一个条件满足，那么 condTermFlagN 被设置成等于 0；
  - mbAddrN is not available
  - mbAddrN 不可用；
  - ctxIdxOffset is equal to 0 and mb_type for the macroblock mbAddrN is equal to SI
  - ctxIdxOffset 等于 0 并且宏块 mbAddrN 的 mb_type 等于 SI；
  - ctxIdxOffset is equal to 3 and mb_type for the macroblock mbAddrN is equal to I_4x4
  - ctxIdxOffset 等于 3 并且宏块 mbAddrN 的 mb_type 等于 I_4x4；
  - ctxIdxOffset is equal to 27 and the macroblock mbAddrN is skipped
  - ctxIdxOffset 等于 27 并且宏块 mbAddrN 没有编码（即 skip 模式）；
  - ctxIdxOffset is equal to 27 and mb_type for the macroblock mbAddrN is equal to B_Direct_16x16
  - ctxIdxOffset 等于 27 并且宏块 mbAddrN 的 mb_type 等于 B_Direct_16x16；
- Otherwise, condTermFlagN is set equal to 1.
- 否则，condTermFlagN 被设置成 1；

The variable ctxIdxInc is derived as

变量 ctxIdxInc 的计算如下：

ctxIdxInc = condTermFlagA + condTermFlagB (9-3)

9.3.3.1.1.4 Derivation process of ctxIdxInc for the syntax element coded_block_pattern

语法元素 coded_block_pattern 的 ctxIdxInc 计算过程

Inputs to this process are ctxIdxOffset and binIdx.

本过程的输入是 ctxIdxOffset 和 binIdx。

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

- If ctxIdxOffset is equal to 73, the following applies
- 如果 ctxIdxOffset 等于 73，使用下面的过程：（前缀部分）

- The derivation process for neighbouring 8x8 luma blocks specified in subclause 6.4.7.2 is invoked with luma8x8BlkIdx = binIdx as input and the output is assigned to mbAddrA, mbAddrB, luma8x8BlkIdxA, and luma8x8BlkIdxB.

  将 luma8x8BlkIdx = binIdx 作为输入，使用 6.4.7.2 小节中的相邻 8x8 亮度块的计算过程，输出赋给 mbAddrA，mbAddrB，luma8x8BlkIdxA 和 luma8x8BlkIdxB。

- Let the variable condTermFlagN (with N being either A or B) be derived as follows.

  变量 condTermFlagN (N 可能为 A 或 B) 的计算如下：

  - If any of the following conditions is true, condTermFlagN is set equal to 0
  - 如果下面的条件满足，那么 condTermFlagN 被设置成等于 0；
    - mbAddrN is not available

      mbAddrN 不可用；

    - mb_type for the macroblock mbAddrN is equal to I_PCM

      宏块 mbAddrN 的 mb_type 等于 I_PCM ；

    - the macroblock mbAddrN is not skipped and ((CodedBlockPatternLuma >> luma8x8BlkIdxN) & 1) is not equal to 0 for the macroblock mbAddrN

      宏块 mbAddrN 没有跳过（skip 模式）并且它的 ((CodedBlockPatternLuma >> luma8x8BlkIdxN) & 1) 不等于 0，

  - Otherwise, condTermFlagN is set equal to 1.
  - 否则，condTermFlagN 被设置成等于 1；

- The variable ctxIdxInc is derived as
- 变量 ctxIdxInc 的计算如下：

$$\text{ctxIdxInc} = \text{condTermFlagA} + 2 * \text{condTermFlagB} \tag{9-4}$$

- Otherwise (ctxIdxOffset is equal to 77), the following applies.
- 否则（ctxIdxOffset 等于 77），使用下面的过程：（后缀部分）
  - The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrA and mbAddrB.

    使用 6.4.7.1 小节中的相邻宏块的计算过程，输出赋给 mbAddrA 和 mbAddrB；

  - Let the variable condTermFlagN (with N being either A or B) be derived as follows.

    变量 condTermFlagN (N 可能为 A 或 B) 的计算如下：

    - If mbAddrN is available and mb_type for the macroblock mbAddrN is equal to I_PCM, condTermFlagN is set equal to 1
    - 如果 mbAddrN 可用并且它的编码模式为 I_PCM，那么 condTermFlagN 被设置成 1 ；
    - If any of the following conditions is true, condTermFlagN is set equal to 0
    - 如果下面的条件满足，那么 condTermFlagN 被设置成 0；
      - mbAddrN is not available or the macroblock mbAddrN is skipped

        mbAddrN 不可用或者宏块 mbAddrN 使用了 skip 编码模式

      - binIdx is equal to 0 and CodedBlockPatternChroma for the macroblock mbAddrN is equal to 0

        binIdx 等于 0 并且宏块 mbAddrN 的 CodedBlockPatternChroma 等于 0 ；

      - binIdx is equal to 1 and CodedBlockPatternChroma for the macroblock mbAddrN is not equal to 2

        binIdx 等于 1 并且宏块 mbAddrN 的 CodedBlockPatternChroma 不等于 2 ；

    - Otherwise, condTermFlagN is set equal to 1.

  - 否则，condTermFlagN 被设置成 1；

- The variable ctxIdxInc is derived as
- 变量 ctxIdxInc 的计算如下

$$\text{ctxIdxInc} = \text{condTermFlagA} + 2 * \text{condTermFlagB} + ((\text{binIdx} == 1) ? 4 : 0) \quad (9\text{-}5)$$

9.3.3.1.1.5 Derivation process of ctxIdxInc for the syntax element mb_qp_delta

语法元素 mb_qp_delta 的计算过程

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

Let prevMbAddr be the macroblock address of the macroblock that precedes the current macroblock in decoding order. When the current macroblock is the first macroblock of a slice, prevMbAddr is marked as not available.

假设 prevMbAddr 为按照解码顺序当前宏块的前一个宏块，如果当前宏块为 slice 中的第一个宏块时，那么 prevMbAddr 被设置成不可用。

- If any of the following conditions is true, ctxIdxInc is set equal to 0
- 如果下面的条件成立，那么 ctxIdxInc 被设置成 0。
  - prevMbAddr is not available or the macroblock prevMbAddr is skipped
  - prevMbAddr 不可用或者宏块 prevMbAddr 使用 skip 编码模式。
  - mb_type of the macroblock prevMbAddr is equal to I_PCM
  - 宏块 prevMbAddr 的 mb_type 等于 I_PCM 。
  - The macroblock prevMbAddr is not coded in Intra_16x16 prediction mode and both CodedBlockPatternLuma and CodedBlockPatternChroma for the macroblock prevMbAddr are equal to 0
  - 宏块 prevMbAddr 的编码模式不是 Intra_16x16 预测模式，并且宏块 prevMbAddr 的 CodedBlockPatternLuma 和 CodedBlockPatternChroma 都等于 0。
  - mb_qp_delta for the macroblock prevMbAddr is equal to 0
  - 宏块 prevMbAddr 的 mb_qp_delta 等于 0；
- Otherwise, ctxIdxInc is set equal to 1.
- 否则，ctxIdxInc 被设置成等于 1。

9.3.3.1.1.6 Derivation process of ctxIdxInc for the syntax elements ref_idx_l0 and ref_idx_l1

语法元素 ref_idx_l0 和 ref_idx_l1 的 ctxIdxInc 计算过程

Inputs to this process are mbPartIdx and the reference picture list suffix lX, where X = 0 or 1.

本过程的输入是 mbPartIdx 和参考图像列表后缀 lX，这里 X = 0 或 1。

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

The derivation process for neighbouring partitions specified in subclause 6.4.7.5 is invoked with mbPartIdx and subMbPartIdx = 0 as input and the output is assigned to mbAddrA\mbPartIdxA and mbAddrB\mbPartIdxB.

将 mbPartIdx 和 subMbPartIdx = 0 作为输入，使用 6.4.7.5 小节中的相邻部分的计算方法，输出赋给 mbAddrA\mbPartIdxA 和 mbAddrB\mbPartIdxB。

Let the variable refIdxZeroFlagN (with N being either A or B) be derived as follows, where ref_idx_lX[mbPartIdxN] specifies the syntax element for the macroblock mbAddrN.

变量 refIdxZeroFlagN (N 可能为 A 或 B) 的计算如下，这里 ref_idx_lX[mbPartIdxN] 表示了宏块 mbAddrN 的语法元素。

- If MbaffFrameFlag is equal to 1, the current macroblock is a frame macroblock, and the macroblock mbAddrN is a field macroblock

  如果 MbaffFrameFlag 等于 1，当前宏块是帧宏块，宏块 mbAddrN 是场宏块

  refIdxZeroFlagN = ((ref_idx_lX[mbPartIdxN] > 1) ? 0 : 1) (9-6)

- Otherwise

  否则

  refIdxZeroFlagN = ((ref_idx_lX[mbPartIdxN] > 0) ? 0 : 1) (9-7)

Let the variable condTermFlagN (with N being either A or B) be derived as follows.

变量 condTermFlagN （N 可能为 A 或者 B）的计算如下：

- If any of the following conditions is true, condTermFlagN is set equal to 0
- 如果下面的任何一个条件满足，那么 condTermFlagN 被设置成等于 0
  - mbAddrN is not available or the macroblock mbAddrN is skipped

    mbAddrN 不可用或者宏块 mbAddrN 使用 skip 编码模式
  - The macroblock mbAddrN is coded in Intra prediction mode

    宏块 mbAddrN 使用帧内预测编码模式
  - mb_type for the macroblock mbAddrN is equal to B_Direct_16x16

    宏块 mbAddrN 的 mb_type 等于 B_Direct_16x16
  - sub_mb_type[mbPartIdxN] for the macroblock mbAddrN is equal to B_Direct_8x8

    宏块 mbAddrN 的 sub_mb_type[mbPartIdxN] 等于 B_Direct_8x8
  - MbPartPredMode(mb_type, mbPartIdxN) is not equal to Pred_LX and not equal to BiPred, where mb_type specifies the syntax element for the macroblock mbAddrN

    MbPartPredMode(mb_type, mbPartIdxN) 不等于 Pred_LX 并且也不等于 BiPred，这里 mb_type 为宏块 mbAddrN 的语法元素
  - refIdxZeroFlagN is equal to 1

    refIdxZeroFlagN 等于 1
- Otherwise, condTermFlagN is set equal to 1.
- 否则，condTermFlagN 等于 1。

The variable ctxIdxInc is derived as

变量 ctxIdxInc 的计算如下：

ctxIdxInc = condTermFlagA + 2 * condTermFlagB (9-8)

注意：这里和最新的协议有出入

9.3.3.1.1.7 Derivation process of ctxIdxInc for the syntax elements mvd_l0 and mvd_l1

语法元素 mvd_l0 和 mvd_l1 的计算过程

Inputs to this process are mbPartIdx, subMbPartIdx, the reference picture list suffix lX, and ctxIdxOffset

本过程的输入是 mbPartIdx，subMbPartIdx，参考图像列表后缀 lX 和 ctxIdxOffset。

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

The derivation process for neighbouring partitions specified in subclause 6.4.7.5 is invoked with mbPartIdx and subMbPartIdx as input and the output is assigned to mbAddrA\mbPartIdxA\subMbPartIdxA and mbAddrB\mbPartIdxB\subMbPartIdxB.

将 mbPartIdx 和 subMbPartIdx 作为输入，使用 6.4.7.5 小节中的相邻部分的计算方法，输出赋给 mbAddrA\mbPartIdxA\subMbPartIdxA 和 mbAddrB\mbPartIdxB\subMbPartIdxB。

Let the variable compIdx be derived as follows.

变量 compIdx 的计算如下：（用于区别运动矢量的）

- If ctxIdxOffset is equal to 40, compIdx is set equal to 0.
- 如果 ctxIdxOffset 等于 40，compIdx 被设置成 0；
- Otherwise (ctxIdxOffset is equal to 47), compIdx is set equal to 1.
- 否则（ctxIdxOffset 等于 47），compIdx 被设置成 1；

Let the variable absMvdCompN (with N being either A or B) be derived as follows.

变量 absMvdCompN （N 可能为 A 或者 B）的计算如下：

- If any of the following conditions is true, absMvdCompN is set equal to 0

  如果下面的任何一个条件成立，那么 absMvdCompN 被设置成等于 0；

  - mbAddrN is not available or the macroblock mbAddrN is skipped

    mbAddrN 不可用或者宏块 mbAddrN 使用 skip 编码模式
  - The macroblock mbAddrN is coded in Intra prediction mode

    宏块 mbAddrN 使用帧内预测编码模式
  - mb_type for the macroblock mbAddrN is equal to B_Direct_16x16

    宏块 mbAddrN 的 mb_type 等于 B_Direct_16x16
  - sub_mb_type[mbPartIdxN] for the macroblock mbAddrN is equal to B_Direct_8x8

    宏块 mbAddrN 的 sub_mb_type[mbPartIdxN] 等于 B_Direct_8x8
  - MbPartPredMode(mb_type, mbPartIdxN) is not equal to Pred_LX and not equal to BiPred, where mb_type specifies the syntax element for the macroblock mbAddrN

    MbPartPredMode(mb_type, mbPartIdxN) 不等于 Pred_LX 并且也不等于 BiPred，这里 mb_type 表示的是宏块 mbAddrN 的语法元素。
- Otherwise, the following applies

  否则，使用下面的过程：

  - If compIdx is equal to 1, MbaffFrameFlag is equal to 1, the current macroblock is a frame macroblock, and the macroblock mbAddrN is a field macroblock
  - 如果 compIdx 等于 1，MbaffFrameFlag 等于 1，当前宏块为帧编码宏块，并且宏块 mbAddrN 是场编码宏块

    absMvdCompN = Abs(mvd_lX[mbPartIdxN][subMbPartIdxN][compIdx]) * 2 (9-9)

  - If compIdx is equal to 1, MbaffFrameFlag is equal to 1, the current macroblock is a field macroblock, and the macroblock mbAddrN is a frame macroblock
  - 如果 compIdx 等于 1，MbaffFrameFlag 等于 1，当前宏块为场编码宏块，并且宏块 mbAddrN 是帧编码宏块

    absMvdCompN = Abs(mvd_lX[mbPartIdxN][subMbPartIdxN][compIdx]) / 2 (9-10)

  - Otherwise
  - 否则

$$absMvdCompN = Abs(mvd\_lX[mbPartIdxN][subMbPartIdxN][compIdx]) \quad (9\text{-}11)$$

The variable ctxIdxInc is derived as follows

变量 ctxIdxInc 的计算如下：

- If (absMvdCompA + absMvdCompB) is less than 3, ctxIdxInc is set equal to 0.

  如果 (absMvdCompA + absMvdCompB) 小于 3，ctxIdxInc 被设置成 0；
- If (absMvdCompA + absMvdCompB) is greater than 32, ctxIdxInc is set equal to 2.

  如果 (absMvdCompA + absMvdCompB) 大于 32，ctxIdxInc 被设置成 2；
- Otherwise ((absMvdCompA + absMvdCompB) is in the range of 3 to 32, inclusive), ctxIdxInc is set equal to 1.

  否则（(absMvdCompA + absMvdCompB) 再 3 到 32 之间，包括这两个数），ctxIdxInc 被设置成 1；

注意：这里和最新的协议有差异

9.3.3.1.1.8 Derivation process of ctxIdxInc for the syntax element intra_chroma_pred_mode

语法元素 intra_chroma_pred_mode 的 ctxIdxInc 的计算过程

Output of this process is c.

本过程的输出是 ctxIdxInc；

The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrA and mbAddrB.

使用 6.4.7.1 小节中的相邻宏块的计算方法，输出赋给 mbAddrA 和 mbAddrB。

Let the variable condTermFlagN (with N being replaced by either A or B) be derived as follows.

变量 condTermFlagN （N 可能是 A 或者 B）的计算如下：

- If any of the following conditions is true, condTermFlagN is set equal to 0
- 如果下面的任何一个条件成立，condTermFlagN 被设置成 0
  - mbAddrN is not available
  - mbAddrN 不可用
  - The macroblock mbAddrN is coded in Inter prediction mode
  - 宏块 mbAddrN 为帧间预测模式
  - mb_type for the macroblock mbAddrN is equal to I_PCM
  - 宏块 mbAddrN 的 mb_type 等于 I_PCM
  - intra_chroma_pred_mode for the macroblock mbAddrN is equal to 0
  - 宏块 mbAddrN 的 intra_chroma_pred_mode 等于 0
- Otherwise, condTermFlagN is set equal to 1.
- 否则，condTermFlagN 被设置成 1；

The variable ctxIdxInc is derived by

变量 ctxIdxInc 计算如下：

$$ctxIdxInc = condTermFlagA + condTermFlagB \quad (9\text{-}12)$$

9.3.3.1.1.9 Derivation process of ctxIdxInc for the syntax element coded_block_flag

语法元素 coded_block_flag 的 ctxIdxInc 计算过程

Input to this process is ctxBlockCat, and

本过程的输入是 ctxBlockCat，和

- If ctxBlockCat is equal to 0, no additional input
- 如果 ctxBlockCat 等于 0，没有其他的输入；
- If ctxBlockCat is equal to 1 or 2, luma4x4BlkIdx
- 如果 ctxBlockCat 等于 1 或者 2，luma4x4BlkIdx；
- If ctxBlockCat is equal to 3, the chroma component index iCbCr
- 如果 ctxBlockCat 等于 3，色差分量索引 iCbCr；
- Otherwise (ctxBlockCat is equal to 4, chroma4x4BlkIdx and the chroma component index compIdx
- 其他（ctxBlockCat 等于 4，chroma4x4BlkIdx 和 色差分量索引 compIdx）；

Output of this process is ctxIdxInc(ctxBlockCat).

本过程的输出是 ctxIdxInc(ctxBlockCat)

Let the variable transBlockN (with N being either A or B) be derived as follows.

变量 transBlockN（N 可能是 A 或者 B）的计算如下：

- If ctxBlockCat is equal to 0, the following applies.
- 如果 ctxBlockCat 等于 0，使用下面的过程
  - The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrN (with N being either A or B).
  - 使用 6.4.7.1 小节中的相邻宏块的计算方法，输出赋给 mbAddrN（N 可能为 A 或者 B）。
  - The variable transBlockN is derived as follows.
  - 变量 transBlockN 的计算如下：
    - If mbAddrN is available and the macroblock mbAddrN is coded in Intra_16x16 prediction mode, the luma DC block of macroblock mbAddrN is assigned to transBlockN
    - 如果 mbAddrN 可用，并且宏块 mbAddrN 使用 Intra_16x16 预测编码模式，那么宏块 mbAddrN 的亮度 DC 块分配给 transBlockN。
    - Otherwise, transBlockN is marked as not available.
    - 否则，transBlockN 被标识为不可用。
- If ctxBlockCat is equal to 1 or 2, the following applies.
- 如果 ctxBlockCat 等于 1 或者 2，使用下面的过程
  - The derivation process for neighbouring 4x4 luma blocks specified in subclause 6.4.7.3 is invoked with luma4x4BlkIdx as input and the output is assigned to mbAddrN, luma4x4BlkIdxN (with N being either A or B).
  - 将 luma4x4BlkIdx 作为输入，使用 6.4.7.3 小节中的方法计算出相邻的 4x4 亮度块，输出赋给 mbAddrN, luma4x4BlkIdxN (N 可能为 A 或 B)。
  - The variable transBlockN is derived as follows.
  - 变量 transBlockN 的计算如下：
    - If mbAddrN is available, the macroblock mbAddrN is not skipped, mb_type for the macroblock mbAddrN is not equal to I_PCM, and (CodedBlockPatternLuma >> (luma4x4BlkIdxN >>2)) is not equal to 0 for the macroblock mbAddrN, the 4x4 luma block with luma4x4BlkIdxN of macroblock mbAddrN is assigned to transBlockN.
    - 如果 mbAddrN 可用，也没有使用 skip 模式编码，宏块 mbAddrN 的 mb_type 也不等于 I_PCM，并且 (CodedBlockPatternLuma >> (luma4x4BlkIdxN >>2) ＆ 1) 不等于 0，那么宏块 mbAddrN 中索引为 luma4x4BlkIdxN 的块赋给 transBlockN。
    - Otherwise, transBlockN is marked as not available.
    - 否则，transBlockN 被标识为不可用。

- If ctxBlockCat is equal to 3, the following applies.
- 如果 ctxBlockCat 等于 3，使用下面的过程
  - The derivation process for neighbouring macroblocks specified in subclause 6.4.7.1 is invoked and the output is assigned to mbAddrN (with N being either A or B).
  - 使用 6.4.7.1 小节中的相邻宏块的计算方法，输出赋给 mbAddrN（N 可能为 A 或者 B）
  - The variable transBlockN is derived as follows.
  - 变量 transBlockN 的计算如下：
    - If mbAddrN is available, the macroblock mbAddrN is not skipped, mb_type for the macroblock mbAddrN is not equal to I_PCM, and CodedBlockPatternChroma is not equal to 0 for the macroblock mbAddrN, the chroma DC block of chroma component iCbCr of macroblock mbAddrN is assigned to transBlockN.
    - 如果 mbAddrN 可用，宏块 mbAddrN 不是 skip 编码模式，宏块 mbAddrN 的 mb_type 不是 I_PCM，并且 CodedBlockPatternChroma 不等于 0，那么宏块 mbAddrN 的色差分量 iCbCr 的 DC 系数块分配给 transBlockN。
    - Otherwise, transBlockN is marked as not available.
    - 否则，transBlockN 被标识为不可用。
- Otherwise (ctxBlockCat is equal to 4), the following applies.
- 否则（ ctxBlockCat 等于 4），使用下面的过程
  - The derivation process for neighbouring 4x4 chroma blocks specified in subclause 6.4.7.4 is invoked with chroma4x4BlkIdx as input and the output is assigned to mbAddrN, chroma4x4BlkIdxN (with N being either A or B).
  - 将 chroma4x4BlkIdx 作为输入，使用 6.4.7.4 小节中的相邻 4x4 色差块的计算方法，输出赋给 mbAddrN, chroma4x4BlkIdxN (N 可能为 A 或者 B)
  - The variable transBlockN is derived as follows.
  - 变量 transBlockN 的计算如下：
    - If mbAddrN is available, the macroblock mbAddrN is not skipped, mb_type for the macroblock mbAddrN is not equal to I_PCM, and CodedBlockPatternChroma is equal to 2 for the macroblock mbAddrN, the 4x4 chroma block with chroma4x4BlkIdxN of the chroma component iCbCr of macroblock mbAddrN is assigned to transBlockN.
    - 如果 mbAddrN 可用，宏块 mbAddrN 没有使用 skip 模式编码，宏块 mbAddrN 的 mb_type 也不等于 I_PCM，并且 CodedBlockPatternChroma 等于 2，那么宏块 mbAddrN 的色差分量 iCbCr 的索引为 chroma4x4BlkIdxN 的色差块分配给 transBlockN。
    - Otherwise, transBlockN is marked as not available.
    - 否则，transBlockN 被标识为不可用。

Let the variable condTermFlagN (with N being either A or B) be derived as follows.

变量 condTermFlagN（N 可能为 A 或者 B）的计算如下：

- If any of the following conditions is true, condTermFlagN is set equal to 0

  如下面的任何一个条件成立，condTermFlagN 设置成 0。
  - mbAddrN is not available and the current macroblock is coded in Inter prediction mode
  - mbAddrN 不可用并且当前宏块是帧间预测模式
  - mbAddrN is available and transBlockN is not available and mb_type for the macroblock mbAddrN is not equal to I_PCM
  - mbAddrN 可用，但是 transBlockN 不可用，宏块 mbAddrN 的 mb_type 也不等于 I_PCM。

- The current macroblock is coded in Intra prediction mode, constrained_intra_pred_flag is equal to 1, the macroblock mbAddrN is available and coded in Inter prediction mode, and slice data partitioning is in use (nal_unit_type is in the range of 2 through 4, inclusive).
- 当前宏块是帧内预测模式，constrained_intra_pred_flag 等于 1，宏块 mbAddrN 可用并且是帧间编码模式，使用 slice 数据分割（nal_unit_typ 的值为 2 到 4，包括 2 和 4）。

- If any of the following conditions is true, condTermFlagN is set equal to 1

  如果下面的任何一个条件成立，condTermFlagN 被设置成 1；

  - mbAddrN is not available and the current macroblock is coded in Intra prediction mode
  - mbAddrN 不可用，当前宏块是帧内预测编码模式
  - mb_type for the macroblock mbAddrN is equal to I_PCM
  - 宏块 mbAddrN 的 mb_type 等于 I_PCM

- Otherwise, condTermFlagN is set equal to the value of the coded_block_flag of the transform block transBlockN that was decoded for the macroblock mbAddrN.
- 否则，condTermFlagN 的值等于宏块 mbAddrN 的变换块 transBlockN 的 coded_block_flag 值。

The variable ctxIdxInc(ctxBlockCat) is derived by

变量 ctxIdxInc(ctxBlockCat) 的计算如下：

$$\text{ctxIdxInc(ctxBlockCat)} = \text{condTermFlagA} + 2 * \text{condTermFlagB} \quad (9\text{-}13)$$

9.3.3.1.2 Assignment process of ctxIdxInc using prior decoded bin values

9.3.3.1.2 利用前面解码的二进制值计算 ctxIdxInc 的过程

Inputs to this process are ctxIdxOffset and binIdx.

本过程的输入是 ctxIdxOffset 和 binIdx

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc

Table 9-31 contains the specification of ctxIdxInc for the given values of ctxIdxOffset and binIdx.

Table 9-31 包含了对于给定的 ctxIdxOffset 和 binIdx 的 ctxIdxInc 说明。

For each value of ctxIdxOffset and binIdx, ctxIdxInc is derived by using some of the values of prior decoded bin values (b_0, b_1, b_2,…, b_k), where the value of the index k is less than the value of binIdx.

对于每一个 ctxIdxOffset 和 binIdx，ctxIdxInc 由前面解码的二进制值 (b_0, b_1, b_2,…, b_k) 计算得到，这里索引 K 的值小于 binIdx 的值

Table 9-31 – Specification of ctxIdxInc for specific values of ctxIdxOffset and binIdx

Value (name) of ctxIdxOffset	binIdx	ctxIdxInc
3	4	(b_3 != 0) ? 5: 6
	5	(b_3 != 0) ? 6: 7
14	2	(b_1 != 1) ? 2: 3
17	4	(b_3 != 0) ? 2: 3
27	2	(b_1 != 0) ? 4: 5
32	4	(b_3 != 0) ? 2: 3
36	2	(b_1 != 0) ? 2: 3

9.3.3.1.3 Assignment process of ctxIdxInc for syntax elements significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1

9.3.3.1.3 对于语法元素 significant_coeff_flag，last_significant_coeff_flag 和 coeff_abs_level_minus1 的 ctxIdxInc 计算过程

Inputs to this process are ctxIdxOffset and binIdx.

本过程的输入是 ctxIdxOffset 和 binIdx。

Output of this process is ctxIdxInc.

本过程的输出是 ctxIdxInc。

The assignment process of ctxIdxInc for syntax elements significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1 as well as for coded_block_flag depends on categories of different blocks denoted by the variable ctxBlockCat. The specification of these block categories is given in Table 9-32.

对于语法元素 significant_coeff_flag，last_significant_coeff_flag， coeff_abs_level_minus1 和 coded_block_flag 的 ctxIdxInc 分配过程和块的类别 ctxBlockCat 有关，这些块的类别的说明见 Table 9-32。

Table 9-32 – Specification of ctxBlockCat for the different blocks

Block description	maxNumCoeff	ctxBlockCat
block of luma DC transform coefficient levels (for macroblock coded in Intra_16x16 prediction mode)	16	0
block of luma AC transform coefficient levels (for macroblock coded in Intra_16x16 prediction mode)	15	1
block of luma transform coefficient levels (for macroblock not coded in Intra_16x16 prediction mode)	16	2
block of chroma DC transform coefficient levels	4	3
block of chroma AC transform coefficient levels	15	4

For the syntax elements significant_coeff_flag and last_significant_coeff_flag the scanning position scanningPos within the regarded block is assigned to ctxIdxInc, where scanningPos ranges from 0 to maxNumCoeff - 2, inclusive:

对于语法元素 significant_coeff_flag 和 last_significant_coeff_flag，在对应块中的扫描位置 scanningPos 分配给 ctxIdxInc，这里 scanningPos 的范围为 0 到 maxNumCoeff – 2，包括这两个数。

ctxIdxInc = scanningPos (9-14)

The scanning position for frame coded blocks relates to the zig-zag scan; the scanning position for field coded blocks relates to the field scan.

帧编码的扫描位置和 zig-zag 扫描有关，场编码块的扫描位置和场扫描有关。

Let numDecodAbsLevelEq1 denotes the accumulated number of decoded transform coefficient levels with absolute value equal to 1, and let numDecodAbsLevelGt1 denotes the accumulated number of decoded transform coefficient levels with absolute value greater than 1. Both numbers are related to the same transform coefficient block, where the current decoding process takes place. Then, for decoding of coeff_abs_level_minus1, ctxIdxInc for coeff_abs_level_minus1 is specified depending on binIdx as follows

假设 numDecodAbsLevelEq1 表示绝对值等于 1 的解码变换系数的个数，numDecodAbsLevelGt1表示绝对值大于 1 的解码变换系数的个数。这两个数和当前解码的变换系数块有关。然后，对于 coeff_abs_level_minus1 的解码，coeff_abs_level_minus1 的 ctxIdxInc 的说明如下，和 binIdx 有关。

- If binIdx is equal to 0, ctxIdxInc is derived by
- 如果 binIdx 等于 0，ctxIdxInc 的计算如下：

ctxIdxInc = ((numDecodAbsLevelGt1 != 0) ? 0: Min(4, 1 + numDecodAbsLevelEq1)) (9-15)

- Otherwise (binIdx is greater than 0), ctxIdxInc is derived by
- 否则（binIdx 大于 0），ctxIdxInc 的计算如下：

ctxIdxInc = 5 + Min(4, numDecodAbsLevelGt1) (9-16)

9.3.3.2 Arithmetic decoding process

9.3.3.2 算术解码过程

Inputs to this process are the bypassFlag, ctxIdx as derived in subclause 9.3.3.1, and the state variables codIRange and codIOffset of the arithmetic decoding engine.

本过程的输入 bypassFlag，在 9.3.3.1 小节中计算得到的 ctxIdx，和算术解码引擎的状态变量 codIRange 和 codIOffset。

Output of this process is the value of the bin.

本过程的输出是 bin 的值。

Figure 9-2 illustrates the whole arithmetic decoding process for a single bin. For decoding the value of a bin, the context index ctxIdx is passed to the arithmetic decoding process DecodeBin(ctxIdx), which is specified as follows.

Figure 9-2 表示了一个比特位的整个算术解码过程。对于解码一个二进制位的值，内容模型索引 ctxIdx 被用来 DecodeBin(ctxIdx)，具体说明如下：

- If bypassFlag is equal to 1, DecodeBypass( ) as specified in subclause 9.3.3.2.3 is invoked.
- 如果 bypassFlag 等于 1，那么使用 9.3.3.2.3 小节中的 DecodeBypass( )。
- If bypassFlag is equal to 0 and ctxIdx is equal to 276, DecodeTerminate( ) as specified in subclause 9.3.3.2.4 is invoked.
- 如果 bypassFlag 等于 0，并且 ctxIdx 等于 276，那么使用 9.3.3.2.4 小节中的 DecodeTerminate( )。
- Otherwise (bypassFlag is equal to 0 and ctxIdx is not equal to 276), DecodeDecision( ) as specified in subclause 9.3.3.2.1 shall be applied.
- 否则（bypassFlag 等于 0，并且 ctxIdx 不等于 276），使用 9.3.3.2.1 小节中的 DecodeDecision( )。

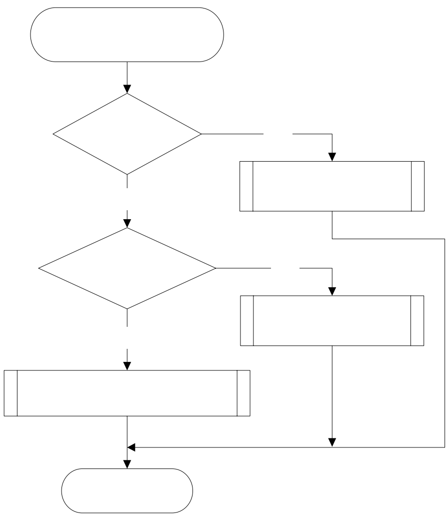


Figure 9-2 – Overview of the arithmetic decoding process for a single bin (informative)

NOTE - Arithmetic coding is based on the principle of recursive interval subdivision. Given a probability estimation p(0) and p(1) = 1 – p(0) of a binary decision (0, 1), an initially given code sub-interval with the range codIRange will be subdivided into two sub-intervals having range p(0) * codIRange and codIRange – p(0) * codIRange, respectively. Depending on the decision, which has been observed, the corresponding sub-interval will be chosen as the new code interval, and a binary code string pointing into that interval will represent the sequence of observed binary decisions. It is useful to distinguish between the most probable symbol (MPS) and the least probable symbol (LPS), so that binary decisions have to be identified as either MPS or LPS, rather than 0 or 1. Given this terminology, each context is specified by the probability p_{LPS} of the LPS and the value of MPS (valMPS), which is either 0 or 1.

注：算术编码原理是基于递归的区间划分，根据二进制(0, 1)的概率估计 p(0) 和 p(1) = 1 – p(0)，给定的初始化子区间 codIRange 被分给为两个子区间 p(0) * codIRange 和 codIRange – p(0) * codIRange，根据二进制的实际值，即编码值，对应的子区间被选为新的子区间，指向这个区间的二进制串就表示了需要编码的数据（这个数据也是二进制的），区别大概率符号(MPS)和小概率符号(LPS)，以至于二进制（0, 1 ）可以被认为是 MPS 或 LPS，而不是 0 或 1 。根据这个术语，每一个内容模型由 LPS 的 p_{LPS} 和 MPS 的值 valMPS来表示，这个值可能是 0 或者 1。

The arithmetic core engine in this Recommendation | International Standard has three distinct properties:

本协议中的算术编码引擎有三个明显的特点：

- The probability estimation is performed by means of a finite-state machine with a table-based transition process between 64 different representative probability states { p_{LPS}(pStateIdx) | 0 <= pStateIdx < 64 } for the LPS probability p_{LPS}. The numbering of the states is arranged in such a way that the probability state with index pStateIdx = 0 corresponds to an LPS probability value of 0.5, with decreasing LPS probability towards higher state indices.
- 概率估计的过程是通过有限状态机来完成的，LPS 的概率 p_{LPS} 是由 64 个不同的典型状态来表示的，{ p_{LPS}(pStateIdx) | 0 <= pStateIdx < 64 }，状态索引 pStateIdx = 0 对应 LPS 的概率为 0.5 ，随着状态索引的增大，LPS 的概率是下降的。
- The range codIRange representing the state of the coding engine is quantised to a small set {Q_1,…,Q_4} of pre-set quantisation values prior to the calculation of the new interval range. Storing a table containing all 64x4 pre-computed product values of Q_i * p_{LPS}(pStateIdx) allows a multiplication-free approximation of the product codIRange * p_{LPS}(pStateIdx).
- 表示算术编码引擎状态的范围 codIRange 在新的区间计算之前由事先设置的量化值 {Q_1,…,Q_4}进行量化，事先保存的 Q_i * p_{LPS}(pStateIdx) 实现了 codIRange * p_{LPS}(pStateIdx) 的无乘法近似。
- For syntax elements or parts thereof for which an approximately uniform probability distribution is assumed to be given a separate simplified encoding and decoding bypass process is used.
- 对于均匀分布的语法元素或部分语法元素采用一种简单的编码和解码方法。

9.3.3.2.1 Arithmetic decoding process for a binary decision

二进制符号的算术解码过程

Inputs to this process are ctxIdx, codIRange, and codIOffset.

本过程的输出是 ctxIdx，codIRange 和 codIOffset。

Outputs of this process are the decoded value binVal, and the updated variables codIRange and codIOffset.

本过程的输出是解码值 binVal，更新的变量 codIRange 和 codIOffset。

Figure 9-3 shows the flowchart for decoding a single decision (DecodeDecision). In a first step, the value of the variable codIRangeLPS is derived as follows.

Figure 9-3 表示了解码一个二进制位的流程，第一步首先是计算变量 codIRangeLPS，如下所示：

Given the current value of codIRange, codIRange is mapped to the index of a quantised value of codIRange, which is denoted by the variable qCodIRangeIdx:

根据 codIRange 的当前值，将 codIRange 映射到 codIRange 的量化值的索引上，用变量 qCodIRangeIdx 来表示：

qCodIRangeIdx =(codIRange >> 6) & 0x03 (9-17)

Given qCodIRangeIdx and pStateIdx associated with ctxIdx, the value of the variable rangeTabLPS as specified in Table 9-33 is assigned to codIRangeLPS:

根据和内容模型 ctxIdx 相关的 qCodIRangeIdx 和 pStateIdx，表 Table 9-33 中的变量 rangeTabLPS 赋给 codIRangeLPS：

codIRangeLPS = rangeTabLPS[pStateIdx][qCodIRangeIdx] (9-18)

In a second step, the value of codIRange – codIRangeLPS is assigned to codIRange, to which the current value of codIOffset is compared. When codIOffset is larger than or equal to codIRange, the value 1 - valMPS is assigned to the decoded value binVal, codIOffset is decremented by codIRange, and codIRange is set equal to codIRangeLPS; otherwise valMPS is assigned to binVal.

第二步是计算新的 codIRange = codIRange – codIRangeLPS，然后用 codIRange 和 codIOffset 进行比较，当 codIOffset 大于或者等于 codIRange 时，那么解码值 binVal 就等于 1 – valMPS，codIOffset = codIOffset – codIRange，同时 codIRange 被设置成 codIRangeLPS；否则 valMPS 被设置成等于 binVal。

Given the decoded value binVal, the state transition is performed as specified in subclause 9.3.3.2.1.1. Depending on the current value of codIRange, renormalization is performed as specified in subclause 9.3.3.2.2.

根据给定的解码值 binVal，然后执行 9.3.3.2.1.1 小节中的状态转移过程，根据当前 codIRange 的值，然后执行 9.3.3.2.2 小节中的归一化过程。

9.3.3.2.1.1 State transition process

状态转移过程

Inputs to this process are the current pStateIdx, the decoded value binVal and valMPS values of the context variable associated with ctxIdx.

本过程的输入是当前的状态索引 pStateIdx，解码值 binVal 和内容模型 ctxIdx 的 valMPS 值。

Outputs of this process are the updated pStateIdx and valMPS of the context variable associated with ctxIdx.

本过程的输出是根新的内容模型 ctxIdx的 pStateIdx 和 valMPS。

Depending on the decoded value binVal, the update of the two variables pStateIdx and valMPS associated with ctxIdx is derived as follows:

根据解码值 binVal，和 ctxIdx 相关的两个变量 pStateIdx 和 valMPS 的计算如下：

```
if( binVal  = =  valMPS )
    pStateIdx = transIdxMPS( pStateIdx )
else {                                                      (9-19)
    if( pStateIdx  = =  0 )
        valMPS = 1 – valMPS
    pStateIdx = transIdxLPS( pStateIdx )
}
```

Table 9-34 specifies the transition rules transIdxMPS( ) and transIdxLPS( ) after decoding the value of valMPS and 1 – valMPS, respectively.

Table 9-34 表示解码完 valMPS 和 1 – valMPS 后 transIdxMPS( ) 和 transIdxLPS( ) 的转移规则。

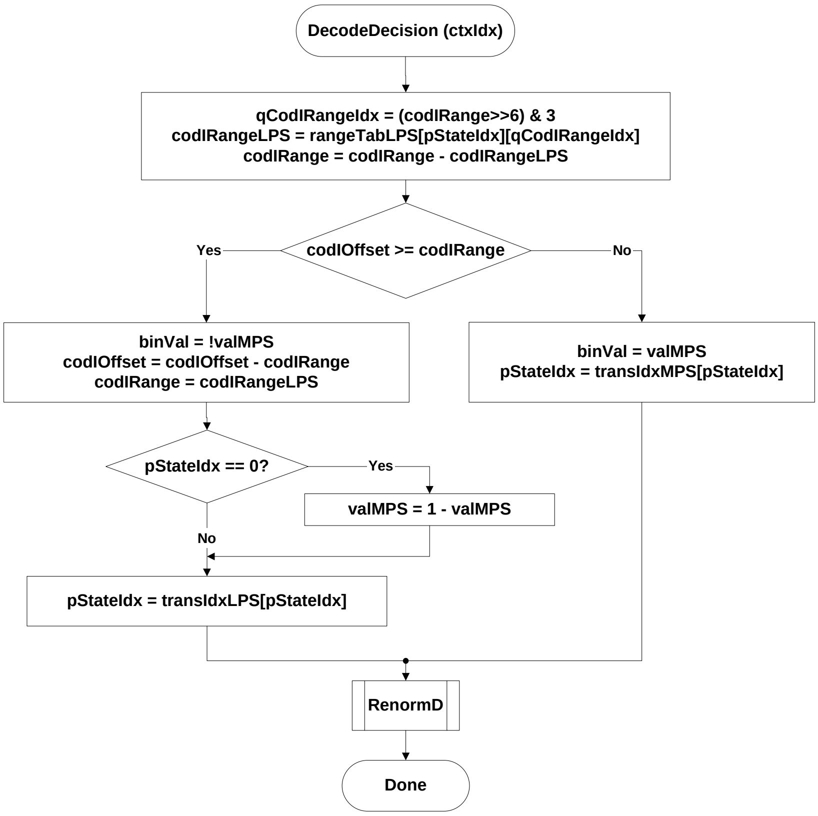


Figure 9-3 – Flowchart for decoding a decision

Table 9-33 – Specification of rangeTabLPS depending on pStateIdx and qCodIRangeIdx

pStateIdx	qCodIRangeIdx				pStateIdx	qCodIRangeIdx			
	0	1	2	3		0	1	2	3
0	128	176	208	240	**32**	27	33	39	45
1	128	167	197	227	**33**	26	31	37	43
2	128	158	187	216	**34**	24	30	35	41
3	123	150	178	205	**35**	23	28	33	39
4	116	142	169	195	**36**	22	27	32	37
5	111	135	160	185	**37**	21	26	30	35
6	105	128	152	175	**38**	20	24	29	33
7	100	122	144	166	**39**	19	23	27	31
8	95	116	137	158	**40**	18	22	26	30
9	90	110	130	150	**41**	17	21	25	28
10	85	104	123	142	**42**	16	20	23	27
11	81	99	117	135	**43**	15	19	22	25
12	77	94	111	128	**44**	14	18	21	24
13	73	89	105	122	**45**	14	17	20	23
14	69	85	100	116	**46**	13	16	19	22
15	66	80	95	110	**47**	12	15	18	21
16	62	76	90	104	**48**	12	14	17	20
17	59	72	86	99	**49**	11	14	16	19
18	56	69	81	94	**50**	11	13	15	18
19	53	65	77	89	**51**	10	12	15	17
20	51	62	73	85	**52**	10	12	14	16
21	48	59	69	80	**53**	9	11	13	15
22	46	56	66	76	**54**	9	11	12	14
23	43	53	63	72	**55**	8	10	12	14
24	41	50	59	69	**56**	8	9	11	13
25	39	48	56	65	**57**	7	9	11	12
26	37	45	54	62	**58**	7	9	10	12
27	35	43	51	59	**59**	7	8	10	11
28	33	41	48	56	**60**	6	8	9	11
29	32	39	46	53	**61**	6	7	9	10
30	30	37	43	50	**62**	6	7	8	9
31	29	35	41	48	**63**	2	2	2	2

Table 9-34 – State transition table

pStateIdx	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
transIdxLPS	0	0	1	2	2	4	4	5	6	7	8	9	9	11	11	12
transIdxMPS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
pStateIdx	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
transIdxLPS	13	13	15	15	16	16	18	18	19	19	21	21	22	22	23	24
transIdxMPS	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
pStateIdx	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**
transIdxLPS	24	25	26	26	27	27	28	29	29	30	30	30	31	32	32	33
transIdxMPS	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
pStateIdx	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**
transIdxLPS	33	33	34	34	35	35	35	36	36	36	37	37	37	38	38	63
transIdxMPS	49	50	51	52	53	54	55	56	57	58	59	60	61	62	62	63

9.3.3.2.2 Renormalization process in the arithmetic decoding engine

算术解码引擎中的归一化处理过程

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

本过程的输入是来自 slice 数据的比特位和变量 codIRange 和 codIOffset。

Outputs of this process are the updated variables codIRange and codIOffset.

本过程的输出是根新的变量 codIRange 和 codIOffset。

A flowchart of the renormalization is shown in Figure 9-4. The current value of codIRange is first compared to 0x0100:

归一化过程的流程如图 Figure 9-4 所示，codIRange 的当前值首先和 0x0100 进行比较：

- If codIRange is larger than or equal to 0x0100, no renormalization is needed and the RenormD process is finished;
- 如果 codIRange 大于或等于 0x0100，那么没有归一化的需求，同时结束归一化过程；
- Otherwise (codIRange is less than 0x0100), the renormalization loop is entered. Within this loop, the value of codIRange is doubled, i.e., left-shifted by 1 and a single bit is shifted into codIOffset by using read_bits(1).
- 否则（codIRange 小于 0x0100），那么就进入归一化的循环中，在这个循环中，codIRange 的值增加一倍，例如左移一位，同时将 read_bits(1) 的值也移进 codIOffset 中。

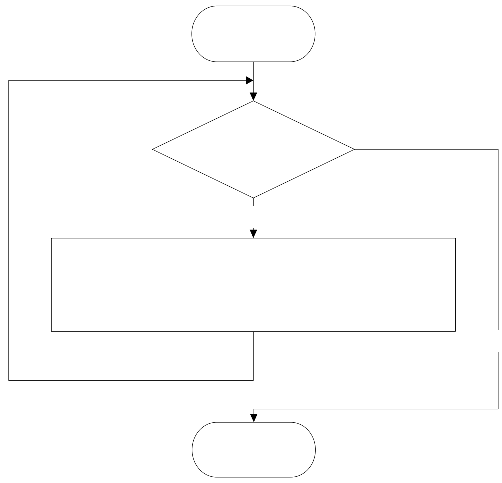


Figure 9-4 – Flowchart of renormalization

9.3.3.2.3 Bypass decoding process for binary decisions

二进制符号的 Bypass 解码过程

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

本过程输入是来自 slice 的数据、变量 codIRange 和 codIOffset。

Outputs of this process are the updated variables codIRange and codIOffset, and the decoded value binVal.

本过程的输出是更新的变量 codIRange 和 codIOffset，以及解码值 binVal。

The bypass decoding process is invoked when bypassFlag is equal to 1. Figure 9-5 shows a flowchart of the corresponding process.

当 bypassFlag 等于 1 时使用 bypass 解码过程，Figure 9-5 表示了相应过程的流程图。

First, the value of codIOffset is doubled, i.e., left-shifted by 1 and a single bit is shifted into codIOffset by using read_bits(1). Then, the value of codIOffset is compared to the value of codIRange.

首先，codIOffset 的值增加一倍，例如左移一位，然后将 read_bits(1) 的数据向左移进 codIOffset 中，codIOffset 的值再和 codIRange 相比较。

- If codIOffset is larger than or equal to codIRange, a value of 1 is assigned to binVal and codIOffset is decremented by codIRange.
- 如果 codIOffset 的值大于或者等于 codIRange，那么 binVal 等于 1，并且 codIOffset 减少 codIRange。
- Otherwise (codIOffset is less than codIRange), a value of 0 is assigned to binVal.
- 否则（codIOffset 小于 codIRange），那么 binVal 等于 0。

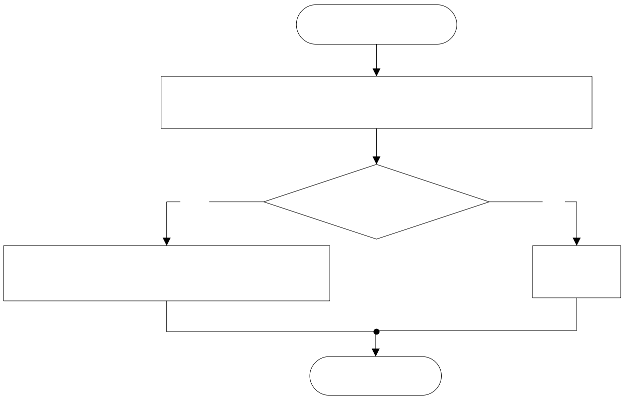


Figure 9-5 – Flowchart of bypass decoding process

9.3.3.2.4 Decoding process for binary decisions before termination

结束前二进制符号的解码过程

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

本过程的输入是来自 slice 的数据，变量 codIRange 和 codIOffset。

Outputs of this process are the updated variables codIRange and codIOffset, and the decoded value binVal.

本过程的输出是更新的 codIRange 和 codIOffset，以及解码值 binVal。

This special decoding routine applies to decoding of end_of_slice_flag and of the bin indicating the I_PCM mode corresponding to ctxIdx equal to 276. Figure 9-6 shows the flowchart of the corresponding decoding process, which is specified as follows.

这个解码方法主要应用于 end_of_slice_flag 的解码和对应于 ctxIdx = 276 表示 I_PCM 编码模式的二进制的解码，Figure 9-6 表示对应的解码过程流程图，具体说明如下：

First, the value of codIRange is decremented by 2. Then, the value of codIOffset is compared to the value of codIRange.

首先，codIRange 的值减去 2，然后 codIOffset 的值和 codIRange 的值进行比较：

- If codIOffset is larger than or equal to codIRange, a value of 1 is assigned to binVal, no renormalization is carried out and CABAC decoding is terminated. In such a case, the last bit inserted in register codIOffset is rbsp_stop_one_bit.
- 如果 codIOffset 大于或者等于 codIRange，那么 binVal 等于 1，也没有归一化过程，同时结束 CABAC 的解码过程，在这种情况下，插入寄存器 codIOffset 中的最后一个比特是 rbsp_stop_one_bit。
- Otherwise (codIOffset is less than codIRange), a value of 0 is assigned to binVal and renormalization is performed as specified in subclause 9.3.3.2.2.
- 否则（codIOffset 小于 codIRange），那么 binVal 等于 0，同时执行 9.3.3.2.2 小节中的归一化过程。

NOTE – This procedure may also be implemented using DecodeDecision(ctxIdx) with ctxIdx = 276. In the case where the decoded value is equal to 1, seven more bits would be read by DecodeDecision(ctxIdx) and a decoding process would have to adjust its bitstream pointer accordingly to properly decode following syntax elements.

注意：这个过程也可以使用 DecodeDecision(ctxIdx) 来实现，ctxIdx = 276 。在这种情况下，解码值等于 1，DecodeDecision(ctxIdx) 读取大于 7 个比特位，根据后面需要解码的语法元素，解码过程必须调整码流的指针。

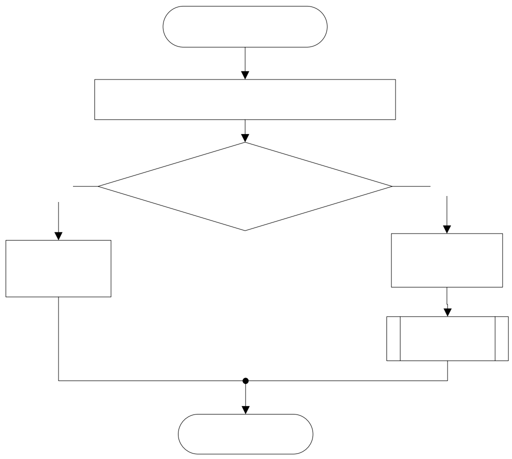


Figure 9-6 – Flowchart of decoding a decision before termination

9.3.4 Arithmetic encoding process (informative)

算术编码过程

This subclause does not form an integral part of this Recommendation | International Standard.

本小节不是本协议的必须部分。

Inputs to this process are decisions that are to be encoded and written.

本过程的输入是需要编码的二进制位。

Outputs of this process are bits that are written to the RBSP.

本过程的输出是写入 RBSP 的比特数据。

This informative subclause describes an arithmetic encoding engine that matches the arithmetic decoding engine described in subclause 9.3.3.2. The encoding engine is essentially symmetric with the decoding engine, i.e., procedures are called in the same order. The following procedures are described in this section: InitEncoder, EncodeDecision, EncodeBypass, EncodeTerminate, which correspond to InitDecoder, DecodeDecision, DecodeBypass, and DecodeTerminate, respectively. The state of the arithmetic encoding engine is represented by a value of the variable codILow pointing to the lower end of a sub-interval and a value of the variable codIRange specifying the corresponding range of that sub-interval.

本小节描述了算术编码引擎，它和 9.3.3.2 小节中的算术解码引擎相对应，编码引擎的处理程序和解码引擎的处理程序是相对称的，在这一部分描述的顺序是：InitEncoder，EncodeDecision，EncodeBypass 和 EncodeTerminate，它们分别对应于 InitDecoder，DecodeDecision，DecodeBypass 和 DecodeTerminate。算术编码引擎的状态由指向子区间下边缘的变量 codILow 和对应子区间的范围变量 codIRange 来表示。

9.3.4.1 Initialisation process for the arithmetic encoding engine (informative)

算术编码引擎的初始化过程

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

This process is invoked before encoding the first macroblock of a slice, and after encoding the pcm_alignment_zero_bit and all pcm_byte data for a macroblock of type I_PCM.

在编码 slice 中的第一个宏块之前和编码完宏块模式为 I_PCM 的 pcm_alignment_zero_bit 和所有的 pcm_byte 数据之后调用本过程。

Outputs of this process are the values codILow, codIRange, firstBitFlag, bitsOutstanding, and symCnt of the arithmetic encoding engine.

本过程的输出是 codILow，codIRange，firstBitFlag，bitsOutstanding 和算术编码引擎的 symCnt。

In the initialisation procedure of the encoder, codILow is set equal to 0, and codIRange is set equal to 0x01FE. Furthermore, a firstBitFlag is set equal to 1, and bitsOutstanding and symCnt counters are set equal to 0.

在编码器的初始化过程中，codILow 被设置成 0，codIRange 被设置成等于 0x01FE，而且 firstBitFlag 设置成1，bitsOutstanding 和 symCnt 计算器设置成 0。

NOTE – The minimum register precision required for codILow is 10 bits and for CodIRange is 9 bits. The precision required for the counters bitsOutstanding and symCnt should be sufficiently large to prevent overflow of the related registers. When MaxBinCountInSlice denotes the maximum total number of binary decisions to encode in one slice, the minimum register precision required for the variables bitsOutstanding and symCnt is given by Ceil(Log2(MaxBinCountInSlice + 1)).

注意：对于 codILow ，最小的寄存器精度为 10 位，CodIRange 为 9 位，bitsOutstanding 和 symCnt 计数器的精度应该足够大以防止相关寄存器的上溢。当 MaxBinCountInSlice 用来表示编码一个 slice 时最大数量的二进制符号时，那么变量 bitsOutstanding 和 symCnt 的最小寄存器精度为 Ceil(Log2(MaxBinCountInSlice + 1))。

9.3.4.2 Encoding process for a binary decision (informative)

二进制数的编码过程

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

Inputs to this process are the context index ctxIdx, the value of binVal to be encoded, and the variables codIRange, codILow and symCnt.

本过程的输入是内容模型索引 ctxIdx，编码的二进制值 binVal，和变量 codIRange，codILow，symCnt。

Outputs of this process are the variables codIRange, codILow, and symCnt.

本过程的输出是变量 codIRange，codILow 和 symCnt。

Figure 9-7 shows the flowchart for encoding a single decision. In a first step, the variable codIRangeLPS is derived as follows.

Figure 9-7 表示了编码一个二进制位的流程图，首先，变量 codIRangeLPS 的计算如下：

Given the current value of codIRange, codIRange is mapped to the index qCodIRangeIdx of a quantised value of codIRange by using Equation 9-17. The value of qCodIRangeIdx and the value of pStateIdx associated with ctxIdx are used to determine the value of the variable rangeTabLPS as specified in Table 9-33, which is assigned to codIRangeLPS. The value of codIRange – codIRangeLPS is assigned to codIRange.

根据当前 codIRange 的值，利用等式 9-17 将 codIRange 映射到 codIRange 量化值的索引上，和 ctxIdx 相关的 qCodIRangeIdx 和 pStateIdx 被用来计算变量 rangeTabLPS 的值，如 Table 9-33 所示，结果赋给变量 codIRangeLPS，codIRange – codIRangeLPS 被赋给 codIRange。

In a second step, the value of binVal is compared to valMPS associated with ctxIdx. When binVal is different from valMPS, codIRange is added to codILow and codIRange is set equal to the value codIRangeLPS. Given the encoded decision, the state transition is performed as specified in subclause 9.3.3.2.1.1. Depending on the current value of codIRange, renormalization is performed as specified in subclause 9.3.4.3. Finally, the variable symCnt is incremented by 1.

第二步，binVal 和 valMPS 进行比较，如果 binVal 和 valMPS 不相同，codIRange 增加 codILow，codIRange 被设置成等于 codIRangeLPS，再根据编码的二进制位，进行状态转移操作，如 9.3.3.2.1.1 小节所示，根据 codIRange 的当前值，进行归一化操作，如 9.3.4.3 小节所示，最后，变量 symCnt 增加 1。

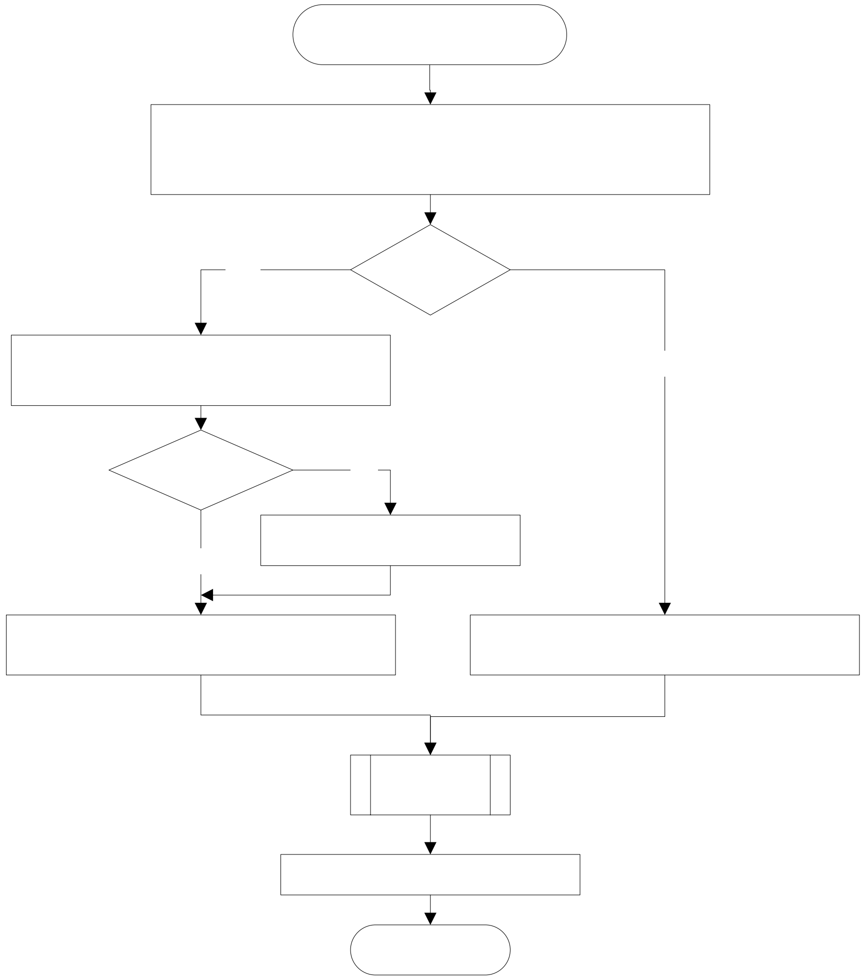


Figure 9-7 – Flowchart for encoding a decision

9.3.4.3 Renormalization process in the arithmetic encoding engine (informative)

算术编码引擎的归一化过程

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

Inputs to this process are the variables codIRange, codILow, firstBitFlag, and bitsOutstanding.

本过程的输入是变量 codIRange，codILow，firstBitFlag 和 bitsOutstanding。

Outputs of this process are zero or more bits written to the RBSP and the updated variables codIRange, codILow, firstBitFlag, and bitsOutstanding.

本过程的输出是写入 RBSP 的比特位和更新的变量 codIRange，codILow，firstBitFlag 和 bitsOutstanding。

Renormalization is illustrated in Figure 9-8. The current value of codIRange is first compared to 0x0100.

归一化过程如图 Figure 9-8 所示，首先 codIRange 的当前值和 0x0100 相比。

- If codIRange is larger than or equal to 0x0100, no renormalization is needed and RenormE is finished.
- 如果 codIRange 大于或者等于 0x0100，那么不需要归一化过程，同时结束 RenormE。（新协议中没有这段描述）
- Otherwise (codIRange is less than 0x0100), the renormalization loop is entered. Within this loop, it is first determined whether a 0 or a 1 can safely be output, i.e., there is no possibility for a carry over in a future iteration. Otherwise, the variable bitsOutstanding is incremented by 1. Finally, the values of codILow and codIRange are doubled.

- 否则（codIRange 小于 0x0100），进入归一化循环。在这个循环中，首先决定输出 0 还是 1，
- （新协议中没有这段描述）

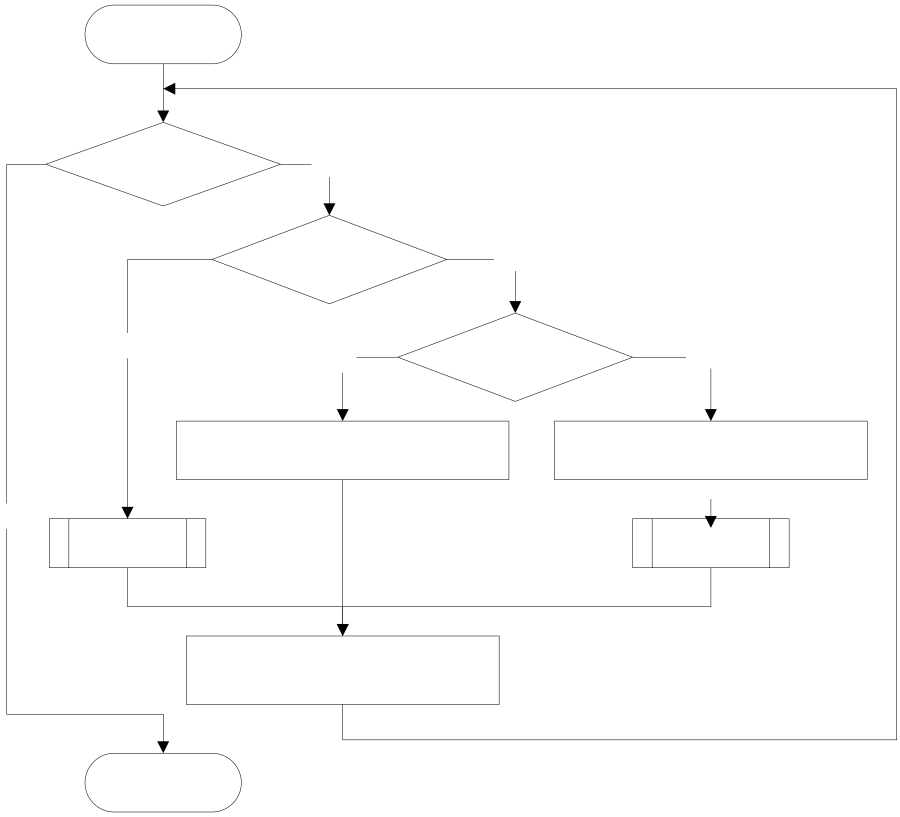


Figure 9-8 – Flowchart of renormalization in the encoder

The PutBit( ) procedure described in Figure 9-9 provides carry over control. It uses the function WriteBits(B, N) that writes N bits with value B to the bitstream and advances the bitstream pointer by N bit positions. This function assumes the existence of a bitstream pointer with an indication of the position of the next bit to be written to the bitstream by the encoding process.

Figure 9-9 描述的 PutBit( ) 过程提供了输出控制，它使用函数 WriteBits(B, N) 写入值为 B 的 N 比特到比特流中，并且码流指针同时也向前移动 N 比特位。这个函数假设码流指针的存在并且指向将要写入下一个比特的位置。

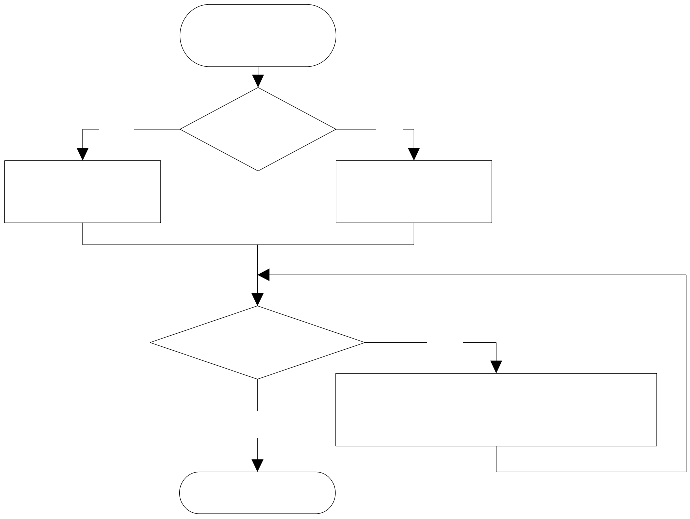


Figure 9-9 – Flowchart of PutBit(B)

9.3.4.4 Bypass encoding process for binary decisions (informative)

二进制符号的 Bypass 编码过程

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

firstBitFl

Inputs to this process are the variables binVal, codILow, codIRange, bitsOutstanding, and symCnt.

本过程的输入变量 binVal，codILow，codIRange，bitsOutstanding 和 symCnt。

Output of this process is a bit written to the RBSP and the updated variables codILow, codIRange, bitsOutstanding, and symCnt.

本过程的输出变量为写入 RBSP 一个比特数据，更新的变量 codILow，codIRange，bitsOutstanding 和 symCnt。

This encoding process applies to all binary decisions with bypassFlag equal to 1. Renormalization is included in the specification of this process as given in Figure 9-10.

当 bypassFlag 等于 1 时使用本编码过程，包含在本过程的归一化过程如 Figure 9-10 所示。

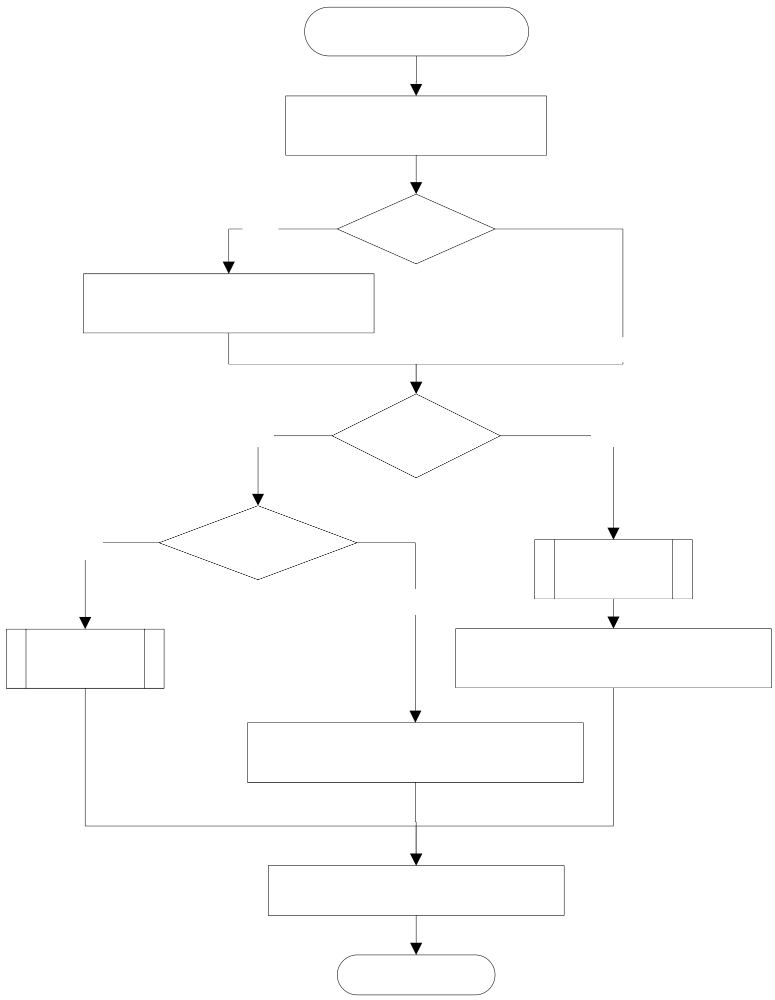


Figure 9-10 – Flowchart of encoding bypass

9.3.4.5 Encoding process for a binary decision before termination (informative)

cod

结束之前二进制符号的编码过程

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

Inputs to this process are the variables binVal, codIRange, codILow, bitsOutstanding, and symCnt.

本过程的输入是变量 binVal，codIRange，codILow，bitsOutstanding 和 symCnt。

Outputs of this process are zero or more bits written to the RBSP and the updated variables codILow, codIRange, bitsOutstanding, and symCnt.

本过程的输出是写入 RBSP 中的零个或多个比特位数据，更新的变量 codILow，codIRange，bitsOutstanding 和 symCnt。

This encoding routine shown in Figure 9-11 applies to encoding of the end_of_slice_flag and of the bin indicating the I_PCM mb_type both associated with ctxIdx equal to 276.

具体的编码过程见 Figure 9-11 所示，该过程主要用来编码 end_of_slice_flag 和 ctxIdx 等于 276 的 mb_type 为 I_PCM 的比特数据。

Yes

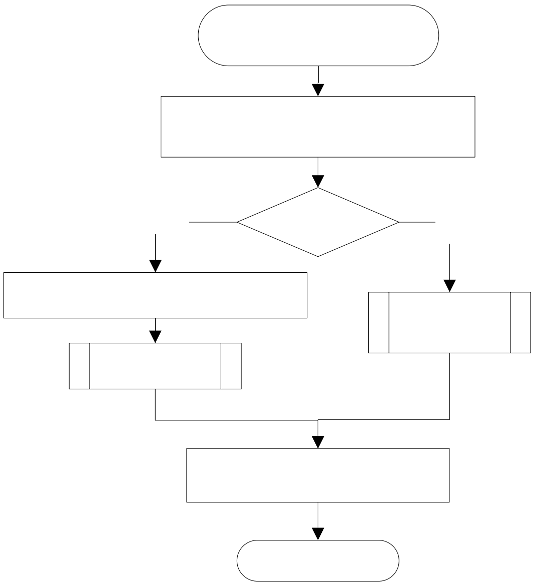


Figure 9-11 – Flowchart of encoding a decision before termination

When the value of binVal to encode is equal to 1 CABAC encoding is terminated and the flushing procedure shown in Figure 9-12 is applied after encoding the end_of_slice_flag. In such a case the last bit written by WriteBits(B, N) contains the rbsp_stop_one_bit.

当 binVal 的值等于 1 ，那么结束 CABAC 的编码过程，在编码完 end_of_slice_flag 后使用图 Figure 9-12 中的 flushing 过程。在这种情况下，WriteBits(B, N) 最后写入的比特位为 rbsp_stop_one_bit。

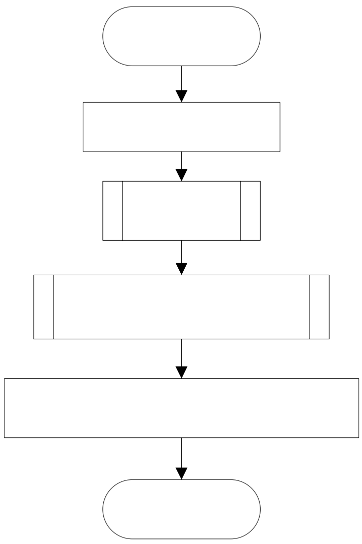


Figure 9-12 – Flowchart of flushing at termination

9.3.4.6 Byte stuffing process (informative)

This subclause does not form an integral part of this Recommendation | International Standard.

这一小节不是本协议的必须部分。

This process is invoked after encoding the last macroblock of the last slice of a picture and after encapsulation.

Inputs to this process are the number of bytes NumBytesInVclNALunits of all VCL NAL units of a picture, the number of macroblocks PicSizeInMbs in the picture, and the number of binary symbols BinCountsInNALunits resulting from encoding the contents of all VCL NAL units of the picture.

Outputs of this process are zero or more bytes appended to the NAL unit.

Let the variable k be set equal to Ceil((Ceil((3 * BinCountsInNALunits – 3 * 96 * PicSizeInMbs) / 32) – NumBytesInVclNALunits) / 3).

- If k is less than or equal to 0, no cabac_zero_word is appended to the NAL unit.
- Otherwise (k is larger than 0), the 3-byte sequence 0x000003 is appended k times to the NAL unit after encapsulation, where the first two bytes 0x0000 represent a cabac_zero_word and the third byte 0x03 represents an emulation_prevention_three_byte.

Annex A
Profiles and levels

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

Profiles and levels specify restrictions on bitstreams and hence limits on the capabilities needed to decode the bitstreams. Profiles and levels may also be used to indicate interoperability points between individual decoder implementations.

NOTE - This Recommendation | International Standard does not include individually selectable "options" at the decoder, as this would increase interoperability difficulties.

Each profile specifies a subset of algorithmic features and limits that shall be supported by all decoders conforming to that profile.

NOTE – Encoders are not required to make use of any particular subset of features supported in a profile.

Each level specifies a set of limits on the values that may be taken by the syntax elements of this Recommendation | International Standard. The same set of level definitions is used with all profiles, but individual implementations may support a different level for each supported profile. For any given profile, levels generally correspond to decoder processing load and memory capability.

A.1 Requirements on video decoder capability

Capabilities of video decoders conforming to this Recommendation | International Standard are specified in terms of the ability to decode video streams conforming to the constraints of profiles and levels specified in this Annex. For each such profile, the level supported for that profile shall also be expressed.

Specific values are specified in this annex for the syntax elements profile_idc and level_idc. All other values of profile_idc and level_idc are reserved for future use by ITU-T | ISO/IEC.

NOTE - Decoders should not infer that if a reserved value of profile_idc or level_idc falls between the values specified in this Recommendation | International Standard that this indicates intermediate capabilities between the specified profiles or levels, as there are no restrictions on the method to be chosen by ITU-T | ISO/IEC for the use of such future reserved values.

A.2 Profiles

A.2.1 Baseline profile

Bitstreams conforming to the Baseline profile shall obey the following constraints:

– Only I and P slice types may be present.
– NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
– Sequence parameter sets shall have frame_mbs_only_flag equal to 1.
– Picture parameter sets shall have weighted_pred_flag and weighted_bipred_idc both equal to 0.
– Picture parameter sets shall have entropy_coding_mode_flag equal to 0.
– Picture parameter sets shall have num_slice_groups_minus1 in the range of 0 to 7, inclusive.
– The level constraints specified for the Baseline profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Baseline profile is specified by profile_idc being equal to 66.

Decoders conforming to the Baseline profile at a specific level shall be capable of decoding all bitstreams in which profile_idc is equal to 66 or constraint_set0_flag is equal to 1 and in which level_idc represents a level less than or equal to the specified level.

A.2.2 Main profile

Bitstreams conforming to the Main profile shall obey the following constraints:

- Only I, P, and B slice types may be present.
- NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
- Arbitrary slice order is not allowed.
- Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
- Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
- The level constraints specified for the Main profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Main profile is specified by profile_idc being equal to 77.

Decoders conforming to the Main profile at a specified level shall be capable of decoding all bitstreams in which profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and in which level_idc represents a level less than or equal to the specified level.

A.2.3 Extended profile

Bitstreams conforming to the Extended profile shall obey the following constraints:

- Sequence parameter sets shall have direct_8x8_inference_flag equal to 1.
- Picture parameter sets shall have entropy_coding_mode_flag equal to 0.
- Picture parameter sets shall have num_slice_groups_minus1 in the range of 0 to 7, inclusive.
- The level constraints specified for the Extended profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Extended profile is specified by profile_idc being equal to 88.

Decoders conforming to the Extended profile at a specified level shall be capable of decoding all bitstreams in which profile_idc is equal to 88 or constraint_set2_flag is equal to 1 and in which level_idc represents a level less than or equal to specified level.

Decoders conforming to the Extended profile at a specified level shall also be capable of decoding all bitstreams in which profile_idc is equal to 66 or constraint_set0_flag is equal to 1, in which level_idc represents a level less than or equal to the specified level.

A.3 Levels

The following is specified for expressing the constraints in this Annex.

- Let access unit n be the n-th access unit in decoding order with the first access unit being access unit 0.
- Let picture n be the primary coded picture or the corresponding decoded picture of access unit n.

A.3.1 Profile-independent level limits

Let the variable fR be derived as follows.

- If picture n is a frame, fR is set equal to $1 \div 172$.
- Otherwise (picture n is a field), fR is set equal to $1 \div (172 * 2)$.

Bitstreams conforming to any profile at a specified level shall obey the following constraints:

a) The nominal removal time of access unit n (with n > 0) from the CPB as specified in subclause C.1.2, satisfies the constraint that $t_{r,n}(n) - t_r(n - 1)$ is greater than or equal to Max(PicSizeInMbs ÷ MaxMBPS, fR), where MaxMBPS is the value specified in Table A-1 that applies to picture n, and PicSizeInMbs is the number of macroblocks in picture n.

b) The difference between consecutive output times of pictures from the DPB as specified in subclause C.2.2, satisfies the constraint that $\Delta t_{o,dpb}(n)$ >= Max(PicSizeInMbs ÷ MaxMBPS, fR), where MaxMBPS is the value specified in Table A-1 for picture n, and PicSizeInMbs is the number of macroblocks of picture n, provided that picture n is a picture that is output and is not the last picture of the bitstream that is output.

c) The sum of the NumBytesInNALunit variables for access unit 0 is less than or equal to 256 * ChromaFormatFactor * (PicSizeInMbs + MaxMBPS * ($t_r(0) - t_{r,n}(0)$)) ÷ MinCR, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture 0 and PicSizeInMbs is the number of macroblocks in picture 0.

d) The sum of the NumBytesInNALunit variables for access unit n (with n > 0) is less than or equal to 256 * ChromaFormatFactor * MaxMBPS * (t_r(n) - t_r(n – 1)) ÷ MinCR, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture n.

e) PicWidthInMbs * FrameHeightInMbs <= MaxFS, where MaxFS is specified in Table A-1

f) PicWidthInMbs <= Sqrt(MaxFS * 8)

g) FrameHeightInMbs <= Sqrt(MaxFS * 8)

h) max_dec_frame_buffering <= MaxDpbSize, where MaxDpbSize is equal to Min(1024 * MaxDPB / (PicWidthInMbs * FrameHeightInMbs * 256 * ChromaFormatFactor), 16) and MaxDPB is given in Table A-1 in units of 1024 bytes. max_dec_frame_buffering is also called DPB size.

i) For the VCL HRD parameters, BitRate[SchedSelIdx] <= 1000 * MaxBR and CpbSize[SchedSelIdx] <= 1000 * MaxCPB for at least one value of SchedSelIdx, where BitRate[SchedSelIdx] is given by Equation E-13 and CpbSize[SchedSelIdx] is given by Equation E-14 when vcl_hrd_parameters_present_flag is equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of 1000 bits/s and 1000 bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1, inclusive. CpbSize[SchedSelIdx] is also called CPB size.

j) For the NAL HRD parameters, BitRate[SchedSelIdx] <= 1200 * MaxBR and CpbSize[SchedSelIdx] <= 1200 * MaxCPB for at least one value of SchedSelIdx, where BitRate[SchedSelIdx] is given by Equation E-13 and CpbSize[SchedSelIdx] is given by Equation E-14 when nal_hrd_parameters_present_flag equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of 1200 bits/s and 1200 bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1.

k) Vertical motion vector component range does not exceed MaxVmvR in units of luma frame samples, where MaxVmvR is specified in Table A-1

l) Horizontal motion vector range does not exceed the range of -2048 to 2047.75, inclusive, in units of luma samples

m) Number of motion vectors per two consecutive macroblocks in decoding order (also applying to the total from the last macroblock of a slice and the first macroblock of the next slice in decoding order) does not exceed MaxMvsPer2Mb, where MaxMvsPer2Mb is specified in Table A-1.

n) Number of bits of macroblock_layer( ) data for any macroblock is not greater than 128 + 2048 * ChromaFormatFactor. Depending on entropy_coding_mode_flag, the bits of macroblock_layer( ) data are counted as follows.

  - If entropy_coding_mode_flag is equal to 0, the number of bits of macroblock_layer( ) data is given by the number of bits in the macroblock_layer( ) syntax structure for a macroblock.

  - Otherwise (entropy_coding_mode_flag is equal to 1), the number of bits of macroblock_layer( ) data for a macroblock is given by the number of times read_bits(1) is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing the macroblock_layer( ) associated with the macroblock.

Table A-1 below gives the limits for each level. Entries marked "-" in Table A-1 denote the absence of a corresponding limit.

Conformance to a particular level shall be specified by setting the syntax element level_idc equal to a value of ten times the level number specified in Table A-1.

Table A-1 – Level limits

Level number	Max macroblock processing rate MaxMBPS (MB/s)	Max frame size MaxFS (MBs)	Max decoded picture buffer size MaxDPB (1024 bytes)	Max video bit rate MaxBR (1000 bits/s or 1200 bits/s)	Max CPB size MaxCPB (1000 bits or 1200 bits)	Vertical MV component range MaxVmvR (luma frame samples)	Min compression ratio MinCR	Max number of motion vectors per two consecutive MBs MaxMvsPer2Mb
1	1 485	99	148.5	64	175	[-64,+63.75]	2	-
1.1	3 000	396	337.5	192	500	[-128,+127.75]	2	-
1.2	6 000	396	891.0	384	1 000	[-128,+127.75]	2	-
1.3	11 880	396	891.0	768	2 000	[-128,+127.75]	2	-
2	11 880	396	891.0	2 000	2 000	[-128,+127.75]	2	-
2.1	19 800	792	1 782.0	4 000	4 000	[-256,+255.75]	2	-
2.2	20 250	1 620	3 037.5	4 000	4 000	[-256,+255.75]	2	-
3	40 500	1 620	3 037.5	10 000	10 000	[-256,+255.75]	2	32
3.1	108 000	3 600	6 750.0	14 000	14 000	[-512,+511.75]	4	16
3.2	216 000	5 120	7 680.0	20 000	20 000	[-512,+511.75]	4	16
4	245 760	8 192	12 288.0	20 000	25 000	[-512,+511.75]	4	16
4.1	245 760	8 192	12 288.0	50 000	62 500	[-512,+511.75]	2	16
4.2	491 520	8 192	12 288.0	50 000	62 500	[-512,+511.75]	2	16
5	589 824	22 080	41 310.0	135 000	135 000	[-512,+511.75]	2	16
5.1	983 040	36 864	69 120.0	240 000	240 000	[-512,+511.75]	2	16

Levels with non-integer level numbers in Table A-1 are referred to as "intermediate levels".

NOTE – All levels have the same status, but some applications may choose to use only the integer-numbered levels.

Informative subclause A.3.3 shows the effect of these limits on frame rates for several example picture formats.

A.3.2 Profile-specific level limits

a) In bitstreams conforming to the Main profile, the removal time of access unit 0 shall satisfy the constraint that the number of slices in picture 0 is less than or equal to $(\text{PicSizeInMbs} + \text{MaxMBPS} * (t_r(0) - t_{r,n}(0))) \div \text{SliceRate}$, where SliceRate is the value specified in Table A-3 that applies to picture 0.

b) In bitstreams conforming to the Main profile, the difference between consecutive removal time of access units n and n - 1 (with n > 0) shall satisfy the constraint that the number of slices in picture n is less than or equal to $\text{MaxMBPS} * (t_r(n) - t_r(n - 1)) \div \text{SliceRate}$, where SliceRate is the value specified in Table A-3 that applies to picture n.

c) In bitstreams conforming to the Main profile, sequence parameter sets shall have direct_8x8_inference_flag equal to 1 for the levels specified in Table A-3.

NOTE – direct_8x8_inference_flag is not relevant to the Baseline profile as it does not allow B slice types (specified in subclause A.2.1), and direct_8x8_inference_flag is equal to 1 for all levels of the Extended profile (specified in subclause A.2.3).

d) In bitstreams conforming to the Main and Extended profiles, sequence parameter sets shall have frame_mbs_only_flag equal to 1 for the levels specified in Table A-3 for the Main profile and in Table A-4 for the Extended profile.

NOTE – frame_mbs_only_flag is equal to 1 for all levels of the Baseline profile (specified in subclause A.2.1).

e) In bitstreams conforming to the Main and Extended profiles, the value of sub_mb_type in B macroblocks shall not be equal to B_Bi_8x4, B_Bi_4x8, or B_Bi_4x4 for the levels in which MinLumaBiPredSize is shown as 8x8 in Table A-3 for the Main profile and in Table A-4 for the Extended profile.

f) In bitstreams conforming to the Baseline and Extended profiles, ($xInt_{max} - xInt_{min} + 6$) * ($yInt_{max} - yInt_{min} + 6$) <= MaxSubMbRectSize in macroblocks coded with mb_type equal to P_8x8, P_8x8ref0 or B_8x8 for all invocations of the process specified in subclause 8.4.2.2.1 used to generate the predicted luma sample array for a single list (list 0 or list 1) for each 8x8 sub-macroblock, where NumSubMbPart(sub_mb_type) > 1, where MaxSubMbRectSize is specified in Table A-2 for the Baseline profile and in Table A-4 for the Extended profile and

   – $xInt_{min}$ as the minimum value of $xInt_L$ among all luma sample predictions for the sub-macroblock
   – $xInt_{max}$ as the maximum value of $xInt_L$ among all luma sample predictions for the sub-macroblock
   – $yInt_{min}$ as the minimum value of $yInt_L$ among all luma sample predictions for the sub-macroblock
   – $yInt_{max}$ as the maximum value of $yInt_L$ among all luma sample predictions for the sub-macroblock

For each level at which a numerical value of MaxSubMbRectSize is specified in Table A-2 for the Baseline profile and in Table A-4 for the Extended profile, the following constraint shall be true for each 8x8 sub-macroblock:

A.3.2.1 Baseline profile limits

Table A-2 specifies limits for each level that are specific to bitstreams conforming to the Baseline profile. Entries marked "-" in Table A-2 denote the absence of a corresponding limit.

Table A-2 – Baseline profile level limits

Level number	MaxSubMbRectSize
1	576
1.1	576
1.2	576
1.3	576
2	576
2.1	576
2.2	576
3	576
3.1	-
3.2	-
4	-
4.1	-
4.2	-
5	-
5.1	-

A.3.2.2 Main profile limits

Table A-3 specifies limits for each level that are specific to bitstreams conforming to the Main profile. Entries marked "-" in Table A-3 denote the absence of a corresponding limit.

Table A-3 – Main profile level limits

Level number	SliceRate	MinLumaBiPredSize	direct_8x8_inference_flag	frame_mbs_only_flag
1	-	-	-	1
1.1	-	-	-	1
1.2	-	-	-	1
1.3	-	-	-	1
2	-	-	-	1
2.1	-	-	-	-
2.2	-	-	-	-
3	22	-	1	-
3.1	60	8x8	1	-
3.2	60	8x8	1	-
4	60	8x8	1	-
4.1	24	8x8	1	-
4.2	24	8x8	1	1
5	24	8x8	1	1
5.1	24	8x8	1	1

A.3.2.3 Extended Profile Limits

Table A-4 specifies limits for each level that are specific to bitstreams conforming to the Extended profile. Entries marked "-" in Table A-4 denote the absence of a corresponding limit.

Table A-4 – Extended profile level limits

Level number	MaxSubMbRectSize	MinLumaBiPredSize	frame_mbs_only_flag
1	576	-	1
1.1	576	-	1
1.2	576	-	1
1.3	576	-	1
2	576	-	1
2.1	576	-	-
2.2	576	-	-
3	576	-	-
3.1	-	8x8	-
3.2	-	8x8	-
4	-	8x8	-
4.1	-	8x8	-
4.2	-	8x8	1
5	-	8x8	1
5.1	-	8x8	1

A.3.3 Effect of level limits on frame rate (informative)

This subclause does not form an integral part of this Recommendation | International Standard.

Table A-5 – Maximum frame rates (frames per second) for some example frame sizes

Level number:					**1**	**1.1**	**1.2**	**1.3**	**2**	**2.1**	**2.2**
Max frame size (macroblocks):					**99**	**396**	**396**	**396**	**396**	**792**	**1 620**
Max macroblocks/second:					**1 485**	**3 000**	**6 000**	**11 880**	**11 880**	**19 800**	**20 250**
Max frame size (samples):					25 344	101 376	101 376	101 376	101 376	202 752	414 720
Max samples/second:					380 160	768 000	1 536 000	3 041 280	3 041 280	5 068 800	5 184 000
Format	**Luma Width**	**Luma Height**	**MBs Total**	**Luma Samples**							
SQCIF	**128**	**96**	48	12 288	30.9	62.5	125.0	172.0	172.0	172.0	172.0
QCIF	**176**	**144**	99	25 344	15.0	30.3	60.6	120.0	120.0	172.0	172.0
QVGA	**320**	**240**	300	76 800	-	10.0	20.0	39.6	39.6	66.0	67.5
525 SIF	**352**	**240**	330	84 480	-	9.1	18.2	36.0	36.0	60.0	61.4
CIF	**352**	**288**	396	101 376	-	7.6	15.2	30.0	30.0	50.0	51.1
525 HHR	**352**	**480**	660	168 960	-	-	-	-	-	30.0	30.7
625 HHR	**352**	**576**	792	202 752	-	-	-	-	-	25.0	25.6
VGA	**640**	**480**	1 200	307 200	-	-	-	-	-	-	16.9
525 4SIF	**704**	**480**	1 320	337 920	-	-	-	-	-	-	15.3
525 SD	**720**	**480**	1 350	345 600	-	-	-	-	-	-	15.0
4CIF	**704**	**576**	1 584	405 504	-	-	-	-	-	-	12.8
625 SD	**720**	**576**	1 620	414 720	-	-	-	-	-	-	12.5
SVGA	**800**	**600**	1 900	486 400	-	-	-	-	-	-	-
XGA	**1024**	**768**	3 072	786 432	-	-	-	-	-	-	-
720p HD	**1280**	**720**	3 600	921 600	-	-	-	-	-	-	-
4VGA	**1280**	**960**	4 800	1 228 800	-	-	-	-	-	-	-
SXGA	**1280**	**1024**	5 120	1 310 720	-	-	-	-	-	-	-
525 16SIF	**1408**	**960**	5 280	1 351 680	-	-	-	-	-	-	-
16CIF	**1408**	**1152**	6 336	1 622 016	-	-	-	-	-	-	-
4SVGA	**1600**	**1200**	7 500	1 920 000	-	-	-	-	-	-	-
1080 HD	**1920**	**1088**	8 160	2 088 960	-	-	-	-	-	-	-
2Kx1K	**2048**	**1024**	8 192	2 097 152	-	-	-	-	-	-	-
4XGA	**2048**	**1536**	12 288	3 145 728	-	-	-	-	-	-	-
16VGA	**2560**	**1920**	19 200	4 915 200	-	-	-	-	-	-	-
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	-	-	-	-	-	-	-
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	-	-	-	-	-	-	-
4Kx2K	**4096**	**2048**	32 768	8 388 608	-	-	-	-	-	-	-
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	-	-	-	-	-	-	-

Table A-5 (continued) – Maximum frame rates (frames per second) for some example frame sizes

Level number:					**3**	**3.1**	**3.2**	**4**	**4.1**	**4.2**
Max frame size (macroblocks):					**1 620**	**3 600**	**5 120**	**8 192**	**8 192**	**8 192**
Max macroblocks/second:					**40 500**	**108 000**	**216 000**	**245 760**	**245 760**	**589 824**
Max frame size (samples):					414 720	921 600	1 310 720	2 097 152	2 097 152	2 097 152
Max samples/second:					10 368 000	27 648 000	55 296 000	62 914 560	62 914 560	125 829 120
Format	**Luma Width**	**Luma Height**	**MBs Total**	**Luma Samples**						
SQCIF	**128**	**96**	48	12 288	172.0	172.0	172.0	172.0	172.0	172.0
QCIF	**176**	**144**	99	25 344	172.0	172.0	172.0	172.0	172.0	172.0
QVGA	**320**	**240**	300	76 800	135.0	172.0	172.0	172.0	172.0	172.0
525 SIF	**352**	**240**	330	84 480	122.7	172.0	172.0	172.0	172.0	172.0
CIF	**352**	**288**	396	101 376	102.3	172.0	172.0	172.0	172.0	172.0
525 HHR	**352**	**480**	660	168 960	61.4	163.6	172.0	172.0	172.0	172.0
625 HHR	**352**	**576**	792	202 752	51.1	136.4	172.0	172.0	172.0	172.0
VGA	**640**	**480**	1 200	307 200	33.8	90.0	172.0	172.0	172.0	172.0
525 4SIF	**704**	**480**	1 320	337 920	30.7	81.8	163.6	172.0	172.0	172.0
525 SD	**720**	**480**	1 350	345 600	30.0	80.0	160.0	172.0	172.0	172.0
4CIF	**704**	**576**	1 584	405 504	25.6	68.2	136.4	155.2	155.2	172.0
625 SD	**720**	**576**	1 620	414 720	25.0	66.7	133.3	151.7	151.7	172.0
SVGA	**800**	**600**	1 900	486 400	-	56.8	113.7	129.3	129.3	172.0
XGA	**1024**	**768**	3 072	786 432	-	35.2	70.3	80.0	80.0	160.0
720p HD	**1280**	**720**	3 600	921 600	-	30.0	60.0	68.3	68.3	136.5
4VGA	**1280**	**960**	4 800	1 228 800	-	-	45.0	51.2	51.2	102.4
SXGA	**1280**	**1024**	5 120	1 310 720	-	-	42.2	48.0	48.0	96.0
525 16SIF	**1408**	**960**	5 280	1 351 680	-	-	-	46.5	46.5	93.1
16CIF	**1408**	**1152**	6 336	1 622 016	-	-	-	38.8	38.8	77.6
4SVGA	**1600**	**1200**	7 500	1 920 000	-	-	-	32.8	32.8	65.5
1080 HD	**1920**	**1088**	8 160	2 088 960	-	-	-	30.1	30.1	60.2
2Kx1K	**2048**	**1024**	8 192	2 097 152	-	-	-	30.0	30.0	60.0
4XGA	**2048**	**1536**	12 288	3 145 728	-	-	-	-	-	-
16VGA	**2560**	**1920**	19 200	4 915 200	-	-	-	-	-	-
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	-	-	-	-	-	-
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	-	-	-	-	-	-
4Kx2K	**4096**	**2048**	32 768	8 388 608	-	-	-	-	-	-
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	-	-	-	-	-	-

Table A-5 (concluded) – Maximum frame rates (frames per second) for some example frame sizes

Level number:					**5**	**5.1**
Max frame size (macroblocks):					**21 696**	**36 864**
Max macroblocks/second:					**589 824**	**983 040**
Max frame size (samples):					5 554 176	9 437 184
Max samples/second:					150 994 944	251 658 240
Format	**Luma Width**	**Luma Height**	**MBs Total**	**Luma Samples**		
SQCIF	**128**	**96**	48	12 288	172.0	172.0
QCIF	**176**	**144**	99	25 344	172.0	172.0
QVGA	**320**	**240**	300	76 800	172.0	172.0
525 SIF	**352**	**240**	330	84 480	172.0	172.0
CIF	**352**	**288**	396	101 376	172.0	172.0
525 HHR	**352**	**480**	660	168 960	172.0	172.0
625 HHR	**352**	**576**	792	202 752	172.0	172.0
VGA	**640**	**480**	1 200	307 200	172.0	172.0
525 4SIF	**704**	**480**	1 320	337 920	172.0	172.0
525 SD	**720**	**480**	1 350	345 600	172.0	172.0
4CIF	**704**	**576**	1 584	405 504	172.0	172.0
625 SD	**720**	**576**	1 620	414 720	172.0	172.0
SVGA	**800**	**600**	1 900	486 400	172.0	172.0
XGA	**1024**	**768**	3 072	786 432	172.0	172.0
720p HD	**1280**	**720**	3 600	921 600	163.8	172.0
4VGA	**1280**	**960**	4 800	1 228 800	122.9	172.0
SXGA	**1280**	**1024**	5 120	1 310 720	115.2	172.0
525 16SIF	**1408**	**960**	5 280	1 351 680	111.7	172.0
16CIF	**1408**	**1152**	6 336	1 622 016	93.1	155.2
4SVGA	**1600**	**1200**	7 500	1 920 000	78.6	131.1
1080 HD	**1920**	**1088**	8 160	2 088 960	72.3	120.5
2Kx1K	**2048**	**1024**	8 192	2 097 152	72.0	120.0
4XGA	**2048**	**1536**	12 288	3 145 728	48.0	80.0
16VGA	**2560**	**1920**	19 200	4 915 200	30.7	51.2
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	27.2	45.3
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	26.7	44.5
4Kx2K	**4096**	**2048**	32 768	8 388 608	-	30.0
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	-	26.7

The following should be noted.

- This Recommendation | International Standard is a variable-frame-size specification. The specific frame sizes in Table A-5 are illustrative examples only.
- As used in Table A-5, "525" refers to typical use for environments using 525 analogue scan lines (of which approximately 480 lines contain the visible picture region), and "625" refers to environments using 625 analogue scan lines (of which approximately 576 lines contain the visible picture region).
- XGA is also known as (aka) XVGA, 4SVGA aka UXGA, 16XGA aka 4Kx3K, CIF aka 625 SIF, 625 HHR aka 2CIF aka half 625 D-1, aka half 625 ITU-R BT.601, 525 SD aka 525 D-1 aka 525 ITU-R BT.601, 625 SD aka 625 D-1 aka 625 ITU-R BT.601.
- Frame rates given are correct for progressive scan modes. The frame rates are also correct for interlaced video coding for the cases of frame height divisible by 32.

Annex B
Byte stream format

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

This annex specifies syntax and semantics of a byte stream format specified for use by application that deliver some or all of the NAL unit stream as an ordered stream of bytes or bits within which the locations of NAL unit boundaries need to be identifiable from patterns in the data, such as ITU-T Recommendation H.222.0 | ISO/IEC 13818-1 systems or ITU-T Recommendation H.320 systems. For bit-oriented delivery, the bit order for the byte stream format is specified to start with the MSB of the first byte, proceed to the LSB of the first byte, followed by the MSB of the second byte, etc.

The byte stream format consists of a sequence of byte stream NAL unit syntax structures. Each byte stream NAL unit syntax structure contains one start code prefix followed by one nal_unit(NumBytesInNALunit) syntax structure. It may (and under some circumstances, it shall) also contain some additional zero_byte syntax elements.

B.1 Byte stream NAL unit syntax and semantics

B.1.1 Byte stream NAL unit syntax

byte_stream_nal_unit(NumBytesInNALunit) {	C	Descriptor
while(next_bits(24) != 0x000001)		
zero_byte /* equal to 0x00 */		f(8)
if(more_data_in_byte_stream( )) {		
start_code_prefix_one_3bytes /* equal to 0x000001 */		f(24)
nal_unit(NumBytesInNALunit)		
}		
}		

B.1.2 Byte stream NAL unit semantics

The order of byte stream NAL units in the byte stream shall follow the decoding order of the NAL units contained in the byte stream NAL units (see subclause 7.4.1.2). The content of each byte stream NAL unit is associated with the same access unit as the NAL unit contained in the byte stream NAL unit (see subclause 7.4.1.2.3).

zero_byte is a single byte equal to 0x00.

When any of the following conditions are fulfilled, the minimum required number of zero_byte syntax elements preceding the start_code_prefix_one_3bytes is equal to 1.

– the nal_unit_type within the nal_unit( ) is equal to 7 (sequence parameter set) or 8 (picture parameter set)

– the byte stream NAL unit syntax structure contains the first NAL unit of an access unit in decoding order, as specified by subclause 7.4.1.2.3.

Any number of additional zero_byte syntax elements may immediately precede the start code prefix within the byte stream NAL unit syntax structure.

start_code_prefix_one_3bytes is a fixed-value sequence of 3 bytes equal to 0x000001. This syntax element is called a start code prefix.

B.2 Byte stream NAL unit decoding process

Input to this process consists of an ordered stream of bytes consisting of a sequence of byte stream NAL unit syntax structures.

Output of this process consists of a sequence of NAL unit syntax structures.

At the beginning of the decoding process, the decoder initialises its current position in the byte stream to the beginning of the byte stream.

The decoder then performs the following step-wise process repeatedly to extract and decode each NAL unit syntax structure in the byte stream:

1. The decoder examines the byte stream, starting at the current position, to detect the location of the next byte-aligned three-byte sequence equal to 0x000001.

   NOTE – This three-byte sequence equal to 0x000001 is a start_code_prefix_one_3bytes syntax element, and all bytes starting at the current position in the byte stream and preceding the start_code_prefix_one_3bytes (if any) are zero_byte syntax elements equal to 0x00.

2. All bytes preceding and including this three-byte sequence are discarded and the current position in the byte stream is set equal to the position of the byte following this three-byte sequence.

3. NumBytesInNALunit is set equal to the number of byte-aligned bytes starting with the byte at the current position in the byte stream up to and including the last byte that precedes the location of any of the following conditions:

   a. A subsequent byte-aligned three-byte sequence equal to 0x000000, or

   b. A subsequent byte-aligned three-byte sequence equal to 0x000001, or

   c. The end of the byte stream, as determined by unspecified means.

4. NumBytesInNALunit bytes are removed from the bitstream and the current position in the byte stream is advanced by NumBytesInNALunit bytes. This sequence of bytes is nal_unit(NumBytesInNALunit) and is decoded using the NAL unit decoding process.

B.3 Decoder byte-alignment recovery (informative)

This subclause does not form an integral part of this Recommendation | International Standard.

Many applications provide data to a decoder in a manner that is inherently byte aligned, and thus have no need for the bit-oriented byte alignment detection procedure described in this subclause.

When a decoder does not have byte alignment with the encoder's byte stream, the decoder may examine the incoming bitstream for the binary pattern '00000000 00000000 00000000 00000001' (31 consecutive bits equal to 0 followed by a bit equal to 1). The bit immediately following this pattern is the first bit of an aligned byte following a start code prefix. Upon detecting this pattern, the decoder will be byte aligned with the encoder and positioned at the start of a NAL unit in the byte stream.

Once byte aligned with the encoder, the decoder can examine the incoming byte stream for subsequent three-byte sequences 0x000001 and 0x000003.

When the three-byte sequence 0x000001 is detected, this is a start code prefix.

When the three-byte sequence 0x000003 is detected, the third byte (0x03) is an emulation_prevention_three_byte to be discarded as specified in subclause 7.4.1.

The byte alignment detection procedure described in this subclause is functionally equivalent to searching a byte sequence for three consecutive zero-valued bytes (0x000000), starting at any alignment position. Detection of this pattern indicates that the next non-zero byte contains the end of a start code prefix (as a conforming byte stream cannot contain more than 23 consecutive zero-valued bits without containing 31 or more consecutive zero-valued bits, allowing detection of 0x000000 relative to any starting alignment position), and the first non-zero bit in that next non-zero byte is the last bit of an aligned byte and is the last bit of a start code prefix.

Annex C
Hypothetical reference decoder

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

This annex specifies the hypothetical reference decoder (HRD) and its use to check bitstream and decoder conformance.

Two types of bitstreams are subject to HRD conformance checking for this Recommendation | International Standard. The first such type of bitstream, called Type I bitstream, is a NAL unit stream containing only the VCL NAL units and filler data NAL units for all access units in the bitstream. The second type of bitstream, called a Type II bitstream, contains, in addition to the VCL NAL units and filler data NAL units for all access units in the bitstream, at least one of the following.

- additional non-VCL NAL units other than filler data NAL units
- all start code prefixes and zero_byte syntax elements that form a byte stream from the NAL unit stream (as specified in Annex B)

Figure C-1 shows the types of HRD conformance checks.

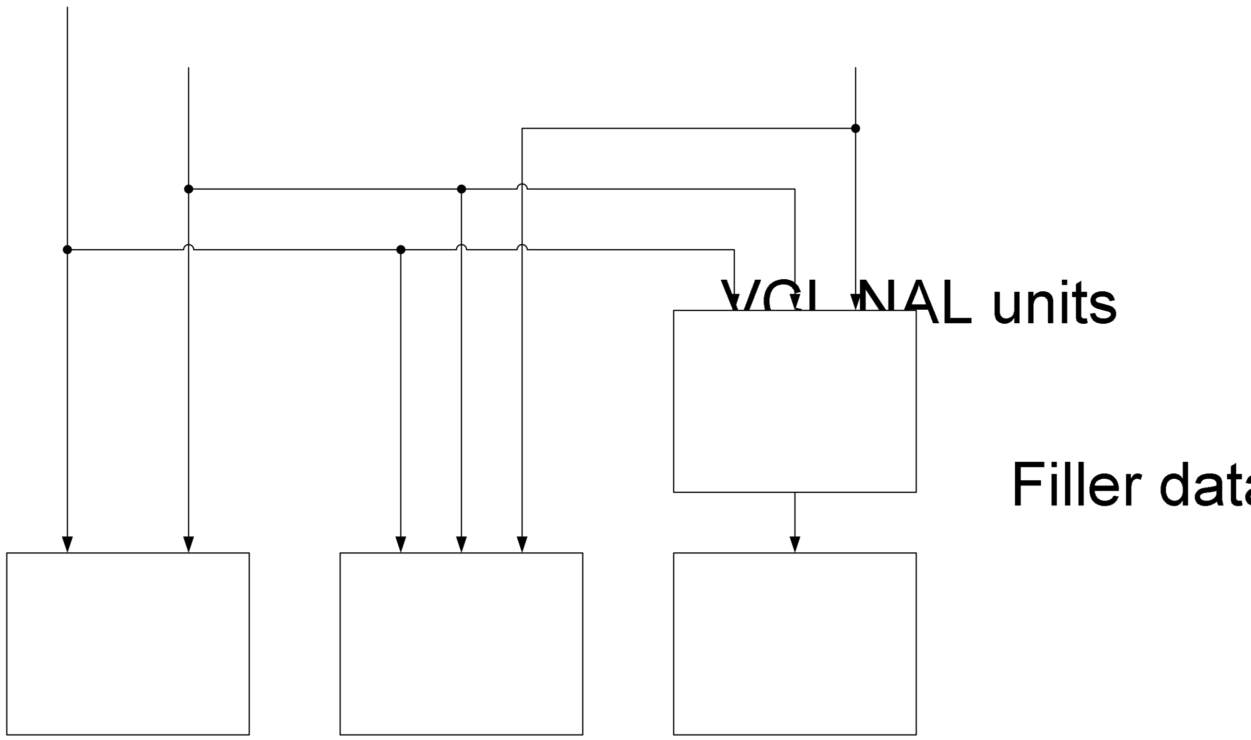


Figure C-1 – Structure of byte streams and NAL unit streams and HRD conformance points

The syntax elements of non-VCL NAL units (or their default values for some of the syntax elements), required for the HRD, are specified in the semantic subclauses of clause 7 and Annexes D and E.

Two types of HRD parameter sets are used. The HRD parameter sets are signalled through video usability information as specified in subclauses E.1 and E.2, which is part of the sequence parameters set syntax structure.

In order to check conformance of a bitstream using the HRD, all sequence parameter sets and picture parameters sets referred to in the VCL NAL units, and corresponding buffering period and picture timing SEI messages shall be conveyed to the HRD, in a timely manner, either in the bitstream (by non-VCL NAL units), or by other means not specified in this Recommendation | International Standard.

In Annexes C, D and E, the specification for "presence" of non-VCL NAL units is also satisfied when those NAL units (or just some of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

NOTE - As an example, synchronization of a non-VCL NAL unit, conveyed by means other than presence in the bitstream, with the NAL units that are present in the bitstream, can be achieved by indicating two points in the bitstream, between which the non-VCL NAL unit would have been present in the bitstream, had the encoder decided to convey it in the bitstream.

When the content of a non-VCL NAL unit is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the non-VCL NAL unit is not required to use the same syntax specified in this annex.

NOTE - When HRD information is contained within the bitstream, it is possible to verify the conformance of a bitstream to the requirements of this subclause based solely on information contained in the bitstream. When the HRD information is not present in the bitstream, as is the case for all "stand-alone" Type I bitstreams, conformance can only be verified when the HRD data is supplied by some other means not specified in this Recommendation | International Standard.

The HRD contains a coded picture buffer (CPB), an instantaneous decoding process, a decoded picture buffer (DPB), and output cropping as shown in Figure C-2.

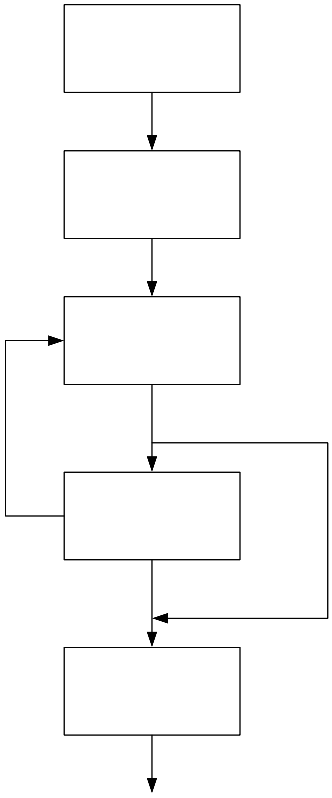


Figure C-2 – HRD buffer model

The CPB size (number of bits) is specified by CpbSize[SchedSelIdx] in Annex E. DPB size (number of frame buffers) is specified by max_dec_frame_buffering in Annex E.

The HRD operates as follows. Data associated with access units that flow into the CPB according to a specified arrival schedule are delivered by the HSS. The data associated with each access unit are removed and decoded instantaneously by the instantaneous decoding process at CPB removal times. Each decoded picture is placed in the DPB at its CPB removal time unless it is output at its CPB removal time and is a non-reference picture. When a picture is placed in the DPB it is removed from the DPB at the later of the DPB output time or the time that it is marked as "unused for reference".

The operation of the CPB is specified in subclause C.1. The instantaneous decoder operation is specified in clauses 8 and 9. The operation of the DPB is specified in subclause C.2. The output cropping is specified in subclause C.2.2.

HSS and HRD information concerning the number of enumerated delivery schedules and their associated bit rates and buffer sizes is specified in subclauses E.1.1, E.1.2, E.2.1 and E.2.2. The HRD is initialised as specified by the buffering period SEI message as specified in subclauses D.1.1 and D.2.1. The removal timing of access units from the CPB and output timing from the DPB are specified in the picture timing SEI message as specified in subclauses D.1.2 and D.2.2. All timing information relating to a specific access unit shall arrive prior to the CPB removal time of the access unit.

The HRD is used to check conformance of bitstreams and decoders as specified in subclauses C.3 and C.4, respectively.

> NOTE - While conformance is guaranteed under the assumption that all frame-rates and clocks used to generate the bitstream match exactly the values signalled in the bitstream, in a real system each of these may vary from the signalled or specified value.

All the arithmetic in this annex is done with real values, so that no rounding errors can propagate. For example, the number of bits in a CPB just prior to or after removal of an access unit is not necessarily an integer.

The variable t_c is derived as follows and is called a clock tick.

$$t_c = \text{num\_units\_in\_tick} \div \text{time\_scale} \qquad \text{(C-1)}$$

Let be(t) and te(b) be the bit equivalent of a time t and the time equivalent of a number of bits b, with the conversion factor being the CPB specified bit rate.

The following is specified for expressing the constraints in this Annex.

- Let access unit n be the n-th access unit in decoding order with the first access unit being access unit 0.
- Let picture n be the primary coded picture or the decoded primary picture of access unit n.

C.1 Operation of coded picture buffer (CPB)

The specifications in this subclause apply independently to each set of CPB parameters that is present and to both Type I and Type II conformance.

C.1.1 Timing of bitstream arrival

The HRD may be initialised at any one of the buffering period SEI messages. Prior to initialisation, the CPB is empty.

NOTE - After initialisation, the HRD is not initialised again by subsequent buffering period SEI messages.

The access unit that is associated with the buffering period SEI message that initializes the CPB is referred to as access unit 0. All other access units are referred to as access unit n with n being incremented by 1 for the next access unit in decoding order.

The time at which the first bit of access unit n begins to enter the CPB is referred to as the initial arrival time $t_{ai}(n)$.

The initial arrival time of access units is derived as follows.

- If the access unit is access unit 0, $t_{ai}(0) = 0$,
- Otherwise (the access unit is access unit n with n > 0), the following applies.
  - If cbr_flag[SchedSelIdx] is equal to 1, the initial arrival time for access unit n, is equal to the final arrival time (which is derived below) of access unit n - 1, i.e.

    $$t_{ai}(n) = t_{af}(n - 1) \qquad (C-2)$$

  - If cbr_flag[SchedSelIdx] is equal to 0 and access unit n is not the first access unit of a subsequent buffering period, the initial arrival time for access unit n is derived by

    $$t_{ai}(n) = \mathrm{Max}(t_{af}(n - 1), t_{ai,earliest}(n)) \qquad (C-3)$$

    where $t_{ai,earliest}(n)$ is given as follows

    $$t_{ai,earliest}(n) = t_{r,n}(n) - (\text{initial\_cpb\_removal\_delay[SchedSelIdx]} + \text{initial\_cpb\_removal\_delay\_offset[SchedSelIdx]}) \div 90000 \qquad (C-4)$$

    with $t_{r,n}(n)$ being the nominal removal time of access unit n from the CPB as specified in subclause C.1.2 and initial_cpb_removal_delay[SchedSelIdx] and initial_cpb_removal_delay_offset[SchedSelIdx] being specified in the previous buffering period SEI message.

  - Otherwise (cbr_flag[SchedSelIdx] is equal to 0 and the subsequent access unit n is the first access unit of a subsequent buffering period), the initial arrival time for the access unit n is derived by

    $$t_{ai}(n) = t_{r,n}(n) - (\text{initial\_cpb\_removal\_delay[SchedSelIdx]} \div 90000) \qquad (C-5)$$

    with initial_cpb_removal_delay[SchedSelIdx] being specified in the buffering period SEI message associated with access unit n.

The final arrival time for access unit n is derived by

$$t_{af}(n) = t_{ai}(n) + b(n) \div \text{BitRate[SchedSelIdx]} \qquad (C-6)$$

where b(n) is the size in bits of access unit n, counting the bits of the Type I bitstream for Type I conformance or the bits of the Type II bitstream for Type II conformance.

The values of SchedSelIdx, BitRate[SchedSelIdx], and CpbSize[SchedSelIdx] are constrained as follows.

- If access unit n and access unit n - 1 are part of different coded video sequences and the content of the active sequence parameter sets of the two coded video sequences differ, the HSS may select a value of BitRate[SchedSelIdx1] or CpbSize[SchedSelIdx1] from among the values of SchedSelIdx1 provided for the coded video sequence containing access unit n that differs from the value of BitRate[SchedSelIdx0] or

CpbSize[SchedSelIdx0] for the value SchedSelIdx0 that was in use for the coded video sequence containing access unit n - 1.

- Otherwise, the HSS continues to operate with the previous values of BitRate[SchedSelIdx0] and CpbSize[SchedSelIdx0].

When the HSS selects values of BitRate[SchedSelIdx] or CpbSize[SchedSelIdx] that differ from those of the previous access unit, the following applies.

- the variable BitRate[SchedSelIdx] comes into effect at time $t_{ai}(n)$
- the variable CpbSize[SchedSelIdx] comes into effect as follows.
  - If the new value of CpbSize[SchedSelIdx] exceeds the old CPB size, it comes into effect at time $t_{ai}(n)$,
  - Otherwise, the new value of CpbSize[SchedSelIdx] comes into effect at the time $t_r(n)$.

C.1.2 Timing of coded picture removal

For access unit 0, the nominal removal time of the access unit from the CPB is specified by

$$t_{r,n}(0) = \text{initial\_cpb\_removal\_delay}[\text{SchedSelIdx}] \div 90000 \tag{C-7}$$

For the first access unit of a buffering period that does not initialise the HRD, the nominal removal time of the access unit from the CPB is specified by

$$t_{r,n}(n) = t_{r,n} (n_b) + t_c * \text{cpb\_removal\_delay}(n) \tag{C-8}$$

where $t_{r,n}(n_b)$ is the nominal removal time of the first picture of the previous buffering period and cpb_removal_delay(n) is specified in the picture timing SEI message associated with access unit n.

When an access unit n is the first access unit of a buffering period, n_b is set equal to n at the removal time of access unit n.

The nominal removal time $t_{r,n}(n)$ of an access unit n that is not the first access unit of a buffering period is given by

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * \text{cpb\_removal\_delay}(n) \tag{C-9}$$

The removal time of access unit n is specified as follows.

- If low_delay_hrd_flag is equal to 0 or $t_{r,n}(n) >= t_{af}(n)$, the removal time of access unit n is specified by

$$t_r(n) = t_{r,n}(n) \tag{C-10}$$

- Otherwise (low_delay_hrd_flag is equal to 1 and $t_{r,n}(n) < t_{af}(n)$), the removal time of access unit n is specified by

$$t_r(n) = t_{r,n}(n) + t_c * \text{Ceil}((t_{af}(n) - t_{r,n}(n)) \div t_c) \tag{C-11}$$

NOTE – The latter case indicates that the size access unit n, b(n), is so large that it prevents removal at the nominal removal time.

C.2 Operation of the decoded picture buffer (DPB)

The decoded picture buffer contains frame buffers. Each of the frame buffers may contain a decoded frame, a decoded complementary field pair or a single (non-paired) decoded field that are marked as "used for reference" (reference pictures) or are held for future output (reordered or delayed pictures). Prior to initialisation, the DPB is empty. The following steps of the subclauses of this subclause all happen instantaneously at $t_r(n)$ and in the sequence listed.

C.2.1 Decoding of gaps in frame_num and storage of "non-existing" frames

If applicable, gaps in frame_num are detected by the decoding process and the resulting frames marked as "non-existing" frames as specified in subclause 8.2.5.2 are inferred and are inserted into the DPB using the sliding window decoded reference picture marking process specified in subclause 8.2.5.3. The DPB fullness is incremented according to the number of additional frames stored in the DPB as a result of the insertion of the "non-existing" frames. If there are not enough empty frame buffers (i.e., DPB size minus DPB fullness is less than the number of "non-existing" frames to be stored), the necessary number of frame buffers is emptied by the sliding window decoded reference picture marking process specified in subclause 8.2.5.3.

C.2.2 Picture decoding and output

Picture n is decoded and its DPB output time $t_{o,dpb}(n)$ is derived by

$$t_{o,dpb}(n) = t_r(n) + t_c * dpb\_output\_delay(n) \quad \text{(C-12)}$$

If $t_{o,dpb}(n) = t_r(n)$, the current picture is output.

NOTE - When the current picture is a reference picture it will be stored in the DPB

Otherwise ($t_{o,dpb}(n) > t_r(n)$), the current picture is output later and will be stored in the DPB (as specified in subclause C.2.4) and is output at time $t_{o,dpb}(n)$ unless indicated not to be output by the decoding or inference of no_output_of_prior_pics_flag equal to 1 at a time that precedes $t_{o,dpb}(n)$.

The output picture shall be cropped, using the cropping rectangle specified in the sequence parameter set for the sequence.

When picture n is a picture that is output and is not the last picture of the bitstream that is output, the value of $\Delta t_{o,dpb}(n)$ is defined as:

$$\Delta t_{o,dpb}(n) = t_{o,dpb}(n_n) - t_{o,dpb}(n) \quad \text{(C-13)}$$

where n_n indicates the picture that follows after picture n in output order.

The decoded picture is temporarily stored (not in the DPB).

C.2.3 Removal of pictures from the DPB before possible insertion of the current picture

The removal of pictures from the DPB before possible insertion of the current picture proceeds as follows.

- If the decoded picture is an IDR picture the following applies.
  - All reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5.
  - When the IDR picture is not the first IDR picture decoded and the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the active sequence parameter set is different from the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the sequence parameter set that was active for the preceding sequence, respectively, no_output_of_prior_pics_flag is inferred to be equal to 1 by the HRD, regardless of the actual value of no_output_of_prior_pics_flag of the active sequence parameter set.

    NOTE - Decoder implementations should try to handle frame or DPB size changes more gracefully than the HRD in regard to changes in PicWidthInMbs or FrameHeightInMbs.
  - When no_output_of_prior_pics_flag is equal to 1 or is inferred to be equal to 1, all frame buffers in the DPB are emptied and DPB fullness is set to 0.
- Otherwise (the decoded picture is not an IDR picture), the following applies.
  - If the slice header of the current picture includes memory_management_control_operation equal to 5, all reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5.
  - Otherwise (the slice header of the current picture does not include memory_management_control_operation equal to 5), the decoded reference picture marking process is invoked as specified in subclause 8.2.5.

All pictures m in the DPB, for which all of the following conditions are true, are removed from the DPB.

- picture m is marked as “unused for reference” or picture m is a non-reference picture. When a picture is a reference frame, it is considered to be marked as "unused for reference" only when both of its fields have been marked as "unused for reference".
- picture m is marked as "non-existing" or its DPB output time is less than or equal to the CPB removal time of the current picture; i.e., $t_{o,dpb}(m) <= t_r(n)$

When a frame or the last field in a frame buffer is removed from the DPB, the DPB fullness is decremented by one.

C.2.4 Current decoded picture marking and storage

C.2.4.1 Marking and storage of a reference decoded picture into the DPB

When the current picture is a reference picture it is stored in the DPB as follows.

- If the current decoded picture is not a second field (in decoding order) of a complementary reference field pair, it is stored into an empty frame buffer and the DPB fullness is incremented by one.

- Otherwise (the current decoded picture is a second field (in decoding order) of a complementary reference field pair), it is stored in the same frame buffer as the previous decoded field.

C.2.4.2 Storage of a non-reference picture into the DPB

A complementary non-reference field pair is specified as follows. When the current picture is a non-reference field, and all of the following conditions apply, the current decoded picture and the previous decoded picture are designated as the second and first fields (respectively) of a complementary non-reference field pair.

- the previous decoded picture (in decoding order) is a non-reference field of opposite parity
- the previous decoded picture (in decoding order) is not a second field of a complementary non-reference field pair
- the previous decoded picture (in decoding order) is still in the DPB

When the current picture is a non-reference picture and current picture n has $t_{o,dpb}(n) > t_r(n)$, it is stored in the DPB as follows.

- If the current decoded picture is not a second field (in decoding order) of a complementary non-reference field pair, it is stored in an empty frame buffer and the DPB fullness is incremented by one.
- Otherwise (the current decoded picture is a second field (in decoding order) of a complementary non-reference field pair), it is stored in the same frame buffer as the first field of the pair.

C.3 Bitstream conformance

A bitstream of coded data conforming to this Recommendation | International Standard fulfils the following requirements.

The bitstream is constructed according to the syntax, semantics, and constraints specified in this Recommendation | International Standard outside of this Annex.

The bitstream is tested by the HRD as specified below:

For Type I bitstreams, the number of tests carried out is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is the syntax element of hrd_parameters( ) following the vcl_hrd_parameters_present_flag or cpb_cnt_minus1 for Type I conformance is determined by the application by other means not specified in this Recommendation | International Standard. One test is carried out for each bit rate and CPB size combination specified by hrd_parameters( ) following the vcl_hrd_parameters_present_flag.

For Type II bitstreams there are two sets of tests. The number of tests of the first set is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is the syntax element of hrd_parameters( ) following the vcl_hrd_parameters_present_flag or cpb_cnt_minus1 for Type II conformance is determined by the application by other means not specified in this Recommendation | International Standard.. One test is carried out for each bit rate and CPB size combination. For these tests, only VCL and filler data NAL units are counted for the input bit rate and CPB storage.

The number of tests of the second set, for Type II bitstreams, is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is the syntax element of hrd_parameters( ) following the nal_hrd_parameters_present_flag or cpb_cnt_minus1 for Type II conformance is determined by the application by other means not specified in this Recommendation | International Standard. One test is carried out for each bit rate and CPB size combination specified by hrd_parameters( ) following the nal_hrd_parameters_present_flag. For these tests, all NAL units (of a Type II NAL unit stream) or all bytes (of a byte stream) are counted for the input bit rate and CPB storage.

For conformant bitstreams, all of the following conditions shall be fulfilled for each of the tests.

- Removal time consistency: For each access unit, the removal times $t_r(n)$ from the CPB computed using different buffering periods SEI messages as starting points for conformance verification shall be as close as possible within the precision of the clocks used (the 90 kHz clock used for initial removal time, and the t_c clock used for subsequent removal time calculations, and BitRate[SchedSelIdx] used for arrival times).

  NOTE – The above condition can be ensured at the encoder by computing the initial CPB removal delay for a buffering period SEI message from the arrival and removal times (subclauses C.1.1 and C.1.2). That is, when the last access unit before the buffering period SEI message is access unit n - 1, the initial CPB removal delay for the next buffering period SEI message should be constrained as specified in the following three equations.

$$\text{initial\_cpb\_removal\_delay[SchedSelIdx]} = 90000 * (t_r(n) - t_r(n - 1) + t_{ai}(n)) \quad \text{(C-14)}$$

If cbr_flag[SchedSelIdx] is equal to 0,

$$\text{initial\_cpb\_removal\_delay[SchedSelIdx]} <= 90000 * (t_r(n) - t_{af}(n - 1)) \quad \text{(C-15)}$$

Otherwise (cbr_flag[SchedSelIdx] is equal to 1),

$$\text{initial\_cpb\_removal\_delay[SchedSelIdx]} = 90000 * (t_r(n) - t_{af}(n - 1)) \quad \text{(C-16)}$$

- CPB underflow and overflow prevention: The CPB shall never overflow or underflow.

  NOTE - In terms of the arrival and removal schedules, this means that, with the exception of some access units in low-delay mode that are described below, all bits from an access unit must be in the CPB at the access unit's nominal removal time $t_{r,n}(n)$. In other words, its final arrival time must be no later than its nominal removal time: $t_{af}(n) \le t_{r,n}(n)$. Further, the nominal removal time $t_{r,n}(n)$ must be no later than the time-equivalent of the buffer size te[CpbSize[SchedSelIdx]]. Note that this prevents both underflow and overflow.

- CPB overflow prevention for big picture removal time: When the final arrival time $t_{af}(n)$ of access unit n to the CPB exceeds its nominal removal time $t_{r,n}(n)$, its size must be such that it can be removed from the buffer without overflow at $t_r(n)$ as specified above.

  NOTE – The final arrival time $t_{af}(n)$ of access unit n to the CPB can only exceed its nominal removal time $t_{r,n}(n)$ if low_delay_hrd_flag is equal to 1.

- Maximum removal rate from the CPB: The nominal removal times of pictures from the CPB (starting from the second picture in decoding order), shall satisfy the constraints on $t_{r,n}(n)$ and $t_r(n)$ expressed in subclauses A.3.1 and A.3.2 for the profile and level specified in the bitstream.

- DPB overflow prevention: Immediately after any decoded picture is added to the DPB, the fullness of the DPB shall be less than or equal to the DPB size as constrained by Annexes A, D, and E for the profile and level specified in the bitstream.

- DPB underflow prevention: All reference pictures shall be present in the DPB when needed for prediction. Each picture shall be present in the DPB at its DPB output time unless it is not stored in the DPB at all, or is removed from the DPB before its output time by one of the processes specified in subclause C.2.

- Maximum output rate from the DPB: The value of $\Delta t_{o,dpb}(n)$ as given by Equation C-13, which is the difference between the output time of a picture and that of the picture immediately following it in output order, shall satisfy the constraint expressed in subclause A.3.1 for the profile and level specified in the bitstream.

C.4 Decoder conformance

A decoder conforming to this Recommendation | International Standard fulfils the following requirements.

A decoder claiming conformance to a specific profile and level shall be able decode successfully all conforming bitstreams specified for decoder conformance in subclause C.3, provided that all sequence parameter sets and picture parameters sets referred to in the VCL NAL units, and appropriate buffering period and picture timing SEI messages are conveyed to the decoder, in a timely manner, either in the bitstream (by non-VCL NAL units), or by external means not specified by this Recommendation | International Standard.

There are two types of conformance that can be claimed by a decoder: output timing conformance and output order conformance.

To check conformance of a decoder, test bitstreams conforming to the claimed profile and level, as specified by subclause C.3 are delivered by a hypothetical stream scheduler (HSS) both to the HRD and to the decoder under test (DUT). The output of both decoders shall be the same. I.e., all pictures output by the HRD are also output by the DUT and all decoded sample values are the same.

For output timing decoder conformance, the HSS operates as described above, with delivery schedules selected only from the subset of values of SchedSelIdx for which the bit rate and CPB size are restricted as specified in Annex A, for the specified profile and level, or with "interpolated" delivery schedules for which the bit rate and CPB size are restricted as specified in Annex A derived from the bit rate and CPB sizes expressed for the provided values of SchedSelIdx as specified below. The same delivery schedule is used for both the HRD and DUT.

When the HRD parameters and the buffering period SEI messages are present with cpb_cnt_minus1 greater than 0, the decoder shall be capable of decoding the bitstream as delivered from the HSS operating using an "interpolated" delivery schedule specified as having peak bit rate r, CPB size c(r), and initial CPB removal delay (f(r) ÷ r) as follows

$$\alpha = (r - \text{BitRate[SchedSelIdx - 1]}) \div (\text{BitRate[SchedSelIdx]} - \text{BitRate[SchedSelIdx - 1]}), \quad \text{(C-17)}$$

$$c(r) = \alpha * \text{CpbSize[SchedSelIdx]} + (1 - \alpha) * \text{CpbSize[SchedSelIdx-1]}, \quad \text{(C-18)}$$

$$\begin{aligned} f(r) = {} & \alpha * \text{initial\_cpb\_removal\_delay[SchedSelIdx]} * \text{BitRate[SchedSelIdx]} + \\ & (1 - \alpha) * \text{initial\_cpb\_removal\_delay[SchedSelIdx - 1]} * \text{BitRate[SchedSelIdx - 1]} \end{aligned} \quad \text{(C-19)}$$

for any SchedSelIdx > 0 and r such that BitRate[SchedSelIdx - 1] <= r <= BitRate[SchedSelIdx - 1] such that r and c(r) are within the limits as specified in Annex A for the maximum bit rate and buffer size for the specified profile and level.

NOTE - initial_cpb_removal_delay[SchedSelIdx] can be different from one buffering period to another and have to be re-calculated.

For output timing decoder conformance, an HRD as described above is used and the timing (relative to the delivery time of the first bit) of picture output is the same for both HRD and the DUT up to a fixed delay.

For output order decoder conformance, the HSS delivers the bitstream to the DUT "by demand" from the DUT, meaning that the HSS delivers bits (in decoding order) only when the DUT requires more bits to proceed with its processing. An HRD as described below is used, and the HSS delivers the bitstream to the HRD by one of the schedules specified in the bitstream or by an "interpolated" schedule such that the bit rate and CPB size are restricted as specified in Annex A. The order of pictures output shall be the same for both HRD and the DUT.

NOTE - This means that for this test, the coded picture buffer of the DUT could be as small as the size of the largest access unit.

For the HRD, the CPB size is equal to CpbSize[SchedSelIdx] for the selected schedule. Removal time from the CPB for the HRD is equal to final bit arrival time and decoding is immediate. The operation of the DPB of this HRD is described below.

C.4.1 Operation of the output order DPB

The decoded picture buffer contains frame buffers. Each of the frame buffers may contain a decoded frame, a decoded complementary field pair or a single (non-paired) decoded field that is marked as "used for reference" or is held for future output (reordered pictures). At HRD initialization, the DPB fullness, measured in frames, is set to 0. The following steps all happen instantaneously when an access unit is removed from the CPB, and in the order listed.

C.4.2 Decoding of gaps in frame_num and storage of "non-existing" pictures

If applicable, gaps in frame_num are detected by the decoding process and the resulting frames marked as "non-existing" frames as specified in subclause 8.2.5.2 are inferred and are inserted into the DPB using the sliding window decoded reference picture marking process specified in subclause 8.2.5.3. The DPB fullness is incremented according to the number of additional frames stored in the DPB as a result of the insertion of the "non-existing" frames. If there are not enough empty frame buffers (i.e., DPB size minus DPB fullness is less than the number of "non-existing" frames to be stored), the necessary number of frame buffers is emptied by the sliding window decoded reference picture marking process specified in subclause 8.2.5.3.

C.4.3 Picture decoding

Primary coded picture n is decoded and is temporarily stored (not in the DPB).

C.4.4 Removal of pictures from the DPB before possible insertion of the current picture

The removal of pictures from the DPB before possible insertion of the current picture proceeds as follows .

- If the decoded picture is an IDR picture the following applies.
  - All reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5.
  - When the IDR picture is not the first IDR picture decoded and the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the active sequence parameter set is different from the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the sequence parameter set that was active for the preceding sequence, respectively, no_output_of_prior_pics_flag is inferred to be equal to 1 by the HRD, regardless of the actual value of no_output_of_prior_pics_flag of the active sequence parameter set.

    NOTE - Decoder implementations should try to handle frame or DPB size changes more gracefully than the HRD in regard to changes in PicWidthInMbs or FrameHeightInMbs.
  - When no_output_of_prior_pics_flag is equal to 1 or is inferred to be equal to 1, all frame buffers in the DPB are emptied and DPB fullness is set to 0.
- Otherwise (the decoded picture is not an IDR picture), the following applies.
  - If the slice header of the current picture includes memory_management_control_operation equal to 5, all reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5.
  - Otherwise (the slice header of the current picture does not include memory_management_control_operation equal to 5), the decoded reference picture marking process is invoked as specified in subclause 8.2.5. Frames marked as "non-existing" and "not used for reference" are emptied and the DPB fullness is decremented by the number of frame buffers emptied.

C.4.5 Current decoded picture marking and storage

C.4.5.1 Storage of picture order counts for the decoded picture

For a decoded frame, TopFieldOrderCnt is stored for the top field and BottomFieldOrderCnt is stored for the bottom field; for a decoded top field, its TopFieldOrderCnt is stored; and for a decoded bottom field, its BottomFieldOrderCnt is stored.

C.4.5.2 Storage and marking of a reference decoded picture into the DPB

If the current decoded picture is a reference picture that is not a second field of a complementary reference field pair, the following operations are performed:

- When there is no empty frame buffer (i.e., DPB fullness is equal to DPB size), one is emptied by the "bumping" process described below.
- The current decoded picture is stored in an empty frame buffer and the DPB fullness is incremented by one.

Otherwise (the current decoded picture is a second field (in decoding order) of a complementary reference field pair), it is stored in the same frame buffer as the first decoded field.

C.4.5.3 Storage and marking of a non-reference decoded picture into the DPB

- If the current decoded picture is a non-reference picture that is not the second field of a complementary non-reference field pair, the following operations are performed:
  - When there is no empty frame buffer (i.e., DPB fullness is equal to DPB size), the following applies
    - If the current picture does not have the lowest value of PicOrderCnt( ) among all pictures in the DPB, a frame buffer is emptied by the "bumping" process described below.
    - Otherwise (the current picture has the lowest value of PicOrderCnt( ) among all pictures in the DPB), the current picture is cropped, using the cropping rectangle specified in the sequence parameter set for the sequence and the cropped picture is output
  - When the current decoded picture has not been output, it is stored in an empty frame buffer and the DPB fullness is incremented by one.
- Otherwise (the current decoded picture is a second field (in decoding order) of a complementary non-reference field pair), it is stored in the same frame buffer as the first decoded field.

C.4.5.4 "Bumping" process

The "bumping" process operates when an empty frame buffer is needed for a decoded (non IDR) picture, as in the following steps:

1. The field marked as "needed for output" that has a lowest value of PicOrderCnt( ) of all fields or frames in the DPB marked as "needed for output", is cropped, using the cropping rectangle specified in the sequence parameter set for the sequence, and the cropped field is output, and the field is marked as "not needed for output".
2. The frame buffer that included the field or frame output in step 1 is checked, and if one of the following conditions is satisfied, the frame buffer is emptied, DPB fullness is decremented and the bumping operation is terminated. Otherwise, steps 1 and 2 are repeated until termination.
   - The frame buffer includes a non-reference non-paired field
   - The frame buffer includes a non-reference frame with both fields marked as "not needed for output"
   - The frame buffer includes a complementary non-reference field pair with both fields marked as "not needed for output".
   - The frame buffer includes a non-paired reference field marked as "unused for reference" and "not needed for output".
   - The frame buffer includes a reference frame with both fields marked as "unused for reference" and "not needed for output".
   - The frame buffer includes a complementary reference field pair with both fields marked as "unused for reference" and "not needed for output".

Annex D
Supplemental enhancement information

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

This annex specifies syntax and semantics for SEI message payloads.

SEI messages assist in processes related to decoding, display or other purposes. However, SEI messages are not required for constructing the luma or chroma samples by the decoding process. Conforming decoders are not required to process this information for output order conformance to this Recommendation | International Standard (see Annex C for the specification of conformance). Some SEI message information is required to check bitstream conformance and for output timing decoder conformance.

In Annex D, specification for presence of SEI messages are also satisfied if those messages (or some subset of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. When present in the bitstream, SEI messages shall obey the syntax and semantics specified in subclauses 7.3.2.3 and 7.4.2.3 and this annex. If the content of an SEI message is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the SEI message is not required to use the same syntax specified in this annex. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

D.1 SEI payload syntax

sei_payload(payloadType, payloadSize) {	**C**	**Descriptor**
if(payloadType = = 0)		
buffering_period(payloadSize)	5	
else if(payloadType = = 1)		
pic_timing(payloadSize)	5	
else if(payloadType = = 2)		
pan_scan_rect(payloadSize)	5	
else if(payloadType = = 3)		
filler_payload(payloadSize)	5	
else if(payloadType = = 4)		
user_data_registered_itu_t_t35(payloadSize)	5	
else if(payloadType = = 5)		
user_data_unregistered(payloadSize)	5	
else if(payloadType = = 6)		
recovery_point(payloadSize)	5	
else if(payloadType = = 7)		
dec_ref_pic_marking_repetition(payloadSize)	5	
else if(payloadType = = 8)		
spare_pic(payloadSize)	5	
else if(payloadType = = 9)		
scene_info(payloadSize)	5	
else if(payloadType = = 10)		
sub_seq_info(payloadSize)	5	
else if(payloadType = = 11)		
sub_seq_layer_characteristics(payloadSize)	5	
else if(payloadType = = 12)		
sub_seq_characteristics(payloadSize)	5	
else if(payloadType = = 13)		
full_frame_freeze(payloadSize)	5	
else if(payloadType = = 14)		
full_frame_freeze_release(payloadSize)	5	
else if(payloadType = = 15)		
full_frame_snapshot(payloadSize)	5	
else if(payloadType = = 16)		
progressive_refinement_segment_start(payloadSize)	5	
else if(payloadType = = 17)		
progressive_refinement_segment_end(payloadSize)	5	
else if(payloadType = = 18)		
motion_constrained_slice_group_set(payloadSize)	5	
else		
reserved_sei_message(payloadSize)	5	
if(!byte_aligned( )) {		
bit_equal_to_one /* equal to 1 */	5	f(1)
while(!byte_aligned( ))		
bit_equal_to_zero /* equal to 0 */	5	f(1)
}		
}		

D.1.1 Buffering period SEI message syntax

buffering_period(payloadSize) {	C	Descriptor
seq_parameter_set_id	5	ue(v)
if(NalHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
if(VclHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
}		

D.1.2 Picture timing SEI message syntax

pic_timing(payloadSize) {	C	Descriptor
if(CpbDpbDelaysPresentFlag) {		
cpb_removal_delay	**5**	u(v)
dpb_output_delay	**5**	u(v)
}		
if(pic_struct_present_flag) {		
pic_struct	**5**	u(4)
for(i = 0; i < NumClockTS ; i++) {		
clock_timestamp_flag[i]	**5**	u(1)
if(clock_timestamp_flag[i]) {		
ct_type	**5**	u(2)
nuit_field_based_flag	**5**	u(1)
counting_type	5	u(5)
full_timestamp_flag	5	u(1)
discontinuity_flag	5	u(1)
cnt_dropped_flag	5	u(1)
n_frames	5	u(8)
if(full_timestamp_flag) {		
seconds_value /* 0..59 */	5	u(6)
minutes_value /* 0..59 */	5	u(6)
hours_value /* 0..23 */	5	u(5)
} else {		
seconds_flag	5	u(1)
if(seconds_flag) {		

seconds_value /* range 0..59 */	5	u(6)
minutes_flag	5	u(1)
if(minutes_flag) {		
minutes_value /* 0..59 */	5	u(6)
hours_flag	5	u(1)
if(hours_flag)		
hours_value /* 0..23 */	5	u(5)
}		
}		
}		
if(time_offset_length > 0)		
time_offset	5	i(v)
}		
}		
}		
}		

D.1.3 Pan-scan rectangle SEI message syntax

pan_scan_rect(payloadSize) {	**C**	**Descriptor**
pan_scan_rect_id	5	ue(v)
pan_scan_rect_cancel_flag	5	u(1)
if(!pan_scan_rect_cancel_flag) {		
pan_scan_cnt_minus1	5	ue(v)
for(i = 0; i <= pan_scan_cnt_minus1; i++) {		
pan_scan_rect_left_offset[i **]**	5	se(v)
pan_scan_rect_right_offset[i **]**	5	se(v)
pan_scan_rect_top_offset[i **]**	5	se(v)
pan_scan_rect_bottom_offset[i **]**	5	se(v)
}		
pan_scan_rect_repetition_period	5	ue(v)
}		
}		

D.1.4 Filler payload SEI message syntax

filler_payload(payloadSize) {	**C**	**Descriptor**
for(k = 0; k < payloadSize; k++)		
ff_byte /* equal to 0xFF */	5	f(8)
}		

D.1.5 User data registered by ITU-T Recommendation T.35 SEI message syntax

user_data_registered_itu_t_t35(payloadSize) {	C	Descriptor
itu_t_t35_country_code	5	b(8)
if(itu_t_t35_country_code != 0xFF)		
i = 1		
else {		
itu_t_t35_country_code_extension_byte	5	b(8)
i = 2		
}		
do {		
itu_t_t35_payload_byte	5	b(8)
i++		
} while(i < payloadSize)		
}		

D.1.6 User data unregistered SEI message syntax

user_data_unregistered(payloadSize) {	C	Descriptor
uuid_iso_iec_11578	**5**	u(128)
for(i = 16; i < payloadSize; i++)		
user_data_payload_byte	5	b(8)
}		

D.1.7 Recovery point SEI message syntax

recovery_point(payloadSize) {	C	Descriptor
recovery_frame_cnt	5	ue(v)
exact_match_flag	5	u(1)
broken_link_flag	5	u(1)
changing_slice_group_idc	5	u(2)
}		

D.1.8 Decoded reference picture marking repetition SEI message syntax

dec_ref_pic_marking_repetition(payloadSize) {	C	Descriptor
original_idr_flag	5	u(1)
original_frame_num	5	ue(v)
if(!frame_mbs_only_flag) {		
original_field_pic_flag	5	u(1)
if(original_field_pic_flag)		
original_bottom_field_flag	5	u(1)
}		
dec_ref_pic_marking( )	5	
}		

D.1.9 Spare picture SEI message syntax

spare_pic(payloadSize) {	C	Descriptor
target_frame_num	5	ue(v)
spare_field_flag	5	u(1)
if(spare_field_flag)		
target_bottom_field_flag	5	u(1)
num_spare_pics_minus1	5	ue(v)
for(i = 0; i < num_spare_pics_minus1 + 1; i++) {		
delta_spare_frame_num[i **]**	5	ue(v)
if(spare_field_flag)		
spare_bottom_field_flag[i **]**	5	u(1)
spare_area_idc[i **]**	5	ue(v)
if(spare_area_idc[i] = = 1)		
for(j = 0; j < PicSizeInMapUnits; j++)		
spare_unit_flag[i **][** j **]**	5	u(1)
else if(spare_area_idc[i] = = 2) {		
mapUnitCnt = 0		
for(j=0; mapUnitCnt < PicSizeInMapUnits; j++) {		
zero_run_length[i **][** j **]**	5	ue(v)
mapUnitCnt += zero_run_length[i][j] + 1		
}		
}		
}		
}		

D.1.10 Scene information SEI message syntax

scene_info(payloadSize) {	C	Descriptor
scene_info_present_flag	5	u(1)
if(scene_info_present_flag) {		
scene_id	5	ue(v)
scene_transition_type	5	ue(v)
if(scene_transition_type > 3)		
second_scene_id	5	ue(v)
}		
}		

D.1.11 Sub-sequence information SEI message syntax

sub_seq_info(payloadSize) {	**C**	**Descriptor**
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
first_ref_pic_flag	5	u(1)
leading_non_ref_pic_flag	5	u(1)
last_pic_flag	5	u(1)
sub_seq_frame_num_flag	5	u(1)
if(sub_seq_frame_num_flag)		
sub_seq_frame_num	5	ue(v)
}		

D.1.12 Sub-sequence layer characteristics SEI message syntax

sub_seq_layer_characteristics(payloadSize) {	**C**	**Descriptor**
num_sub_seq_layers_minus1	5	ue(v)
for(layer = 0; layer <= num_sub_seq_layers_minus1; layer++) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
}		

D.1.13 Sub-sequence characteristics SEI message syntax

sub_seq_characteristics(payloadSize) {	**C**	**Descriptor**
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
duration_flag	5	u(1)
if(duration_flag)		
sub_seq_duration	5	u(32)
average_rate_flag	5	u(1)
if(average_rate_flag) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
num_referenced_subseqs	5	ue(v)
for(n = 0; n < num_referenced_subseqs; n++) {		
ref_sub_seq_layer_num	5	ue(v)
ref_sub_seq_id	5	ue(v)
ref_sub_seq_direction	5	u(1)
}		
}		

D.1.14 Full-frame freeze SEI message syntax

full_frame_freeze(payloadSize) {	C	Descriptor
full_frame_freeze_repetition_period	5	ue(v)
}		

D.1.15 Full-frame freeze release SEI message syntax

full_frame_freeze_release(payloadSize) {	C	Descriptor
}		

D.1.16 Full-frame snapshot SEI message syntax

full_frame_snapshot(payloadSize) {	C	Descriptor
snapshot_id	5	ue(v)
}		

D.1.17 Progressive refinement segment start SEI message syntax

progressive_refinement_segment_start(payloadSize) {	C	Descriptor
progressive_refinement_id	5	ue(v)
num_refinement_steps_minus1	5	ue(v)
}		

D.1.18 Progressive refinement segment end SEI message syntax

progressive_refinement_segment_end(payloadSize) {	C	Descriptor
progressive_refinement_id	5	ue(v)
}		

D.1.19 Motion-constrained slice group set SEI message syntax

motion_constrained_slice_group_set(payloadSize) {	C	Descriptor
num_slice_groups_in_set_minus1	5	ue(v)
for(i = 0; i <= num_slice_groups_in_set_minus1; i++)		
slice_group_id[i **]**	5	u(v)
exact_sample_value_match_flag	5	u(1)
pan_scan_rect_flag	5	u(1)
if(pan_scan_rect_flag)		
pan_scan_rect_id	5	ue(v)
}		

D.1.20 Reserved SEI message syntax

reserved_sei_message(payloadSize) {	C	Descriptor
for(i = 0; i < payloadSize; i++)		
reserved_sei_message_payload_byte	5	b(8)
}		

D.2 SEI payload semantics

D.2.1 Buffering period SEI message semantics

When NalHrdBpPresentFlag or VclHrdBpPresentFlag are equal to 1, a buffering period SEI message can be associated with any access unit in the bitstream, and a buffering period SEI message shall be associated with each IDR access unit and with each access unit associated with a recovery point SEI message.

NOTE – For some applications, the frequent presence of a buffering period SEI message may be desirable.

A buffering period is specified as the set of access units between two instances of the buffering period SEI message in decoding order.

seq_parameter_set_id specifies the sequence parameter set that contains the sequence HRD attributes. The value of seq_parameter_set_id shall be equal to the value of seq_parameter_set_id in the picture parameter set referenced by the primary coded picture associated with the buffering period SEI message. The value of seq_parameter_set_id shall be in the range of 0 to 31, inclusive.

initial_cpb_removal_delay[SchedSelIdx] specifies the delay for the SchedSelIdx-th CPB between the time of arrival in the CPB of the first bit of the coded data associated with the access unit associated with the buffering period SEI message and the time of removal from the CPB of the coded data associated with the same access unit, for the first buffering period after HRD initialisation. The syntax element has a length in bits given by initial_cpb_removal_delay_length_minus1 + 1. It is in units of a 90 kHz clock. initial_cpb_removal_delay shall not be equal to 0 and shall not exceed the time-equivalent of the CPB size.

initial_cpb_removal_delay_offset[SchedSelIdx] is used for the SchedSelIdx-th CPB in combination with the cpb_removal_delay to specify the initial delivery time of coded access units to the CPB. initial_cpb_removal_delay_offset is in units of a 90 kHz clock. The initial_cpb_removal_delay_offset syntax element is a fixed length code whose length in bits is given by initial_cpb_removal_delay_length_minus1 + 1. This syntax element is not used by decoders and is needed only for the delivery scheduler (HSS) specified in Annex C.

Over the entire coded video sequence, the sum of initial_cpb_removal_delay[SchedSelIdx] and initial_cpb_removal_delay_offset[SchedSelIdx] shall be constant for each value of SchedSelIdx.

D.2.2 Picture timing SEI message semantics

When CpbDpbDelaysPresentFlag is equal to 1, a picture timing SEI Message shall be associated with every access unit in the bitstream.

cpb_removal_delay specifies how many clock ticks (see subclause E.2.1) to wait after removal from the CPB of the access unit associated with the most recent buffering period SEI message before removing from the buffer the access unit data associated with the picture timing SEI message. This value is also used to calculate an earliest possible time of arrival of access unit data into the CPB for the HSS, as specified in Annex C. The syntax element is a fixed length code whose length in bits is given by cpb_removal_delay_length_minus1 + 1. The cpb_removal_delay is the remainder of a $2^{(\text{cpb\_removal\_delay\_length\_minus1} + 1)}$ counter.

The value of cpb_removal_delay for the first picture in the bitstream shall be equal to 0.

dpb_output_delay is used to compute the DPB output time of the picture. It specifies how many clock ticks to wait after removal of an access unit from the CPB before the decoded picture can be output from the DPB (see subclause C.2).

NOTE - A picture is not removed from the DPB at its output time if it is still marked as "used for short-term reference" or "used for long-term reference".

NOTE - Only one dpb_output_delay is specified for a decoded picture.

The size of the syntax element dpb_output_delay is given in bits by dpb_output_delay_length_minus1 + 1.

The picture output order established by the values of this syntax element shall be the same order as established by the values of PicOrderCnt( ) of the pictures in the sequence (see subclause C.4.5) except for the following cases:

– For a decoded frame, both of its fields have the same output time but the value of PicOrderCnt( ) for the top and bottom fields may differ.

– For a complementary field pair, the fields have different output times but may have the same value of PicOrderCnt( ).

NOTE - Frame doubling can facilitate the display, for example, of 25p video on a 50p display and 29.97p video on a 59.94p display. Using frame doubling and frame tripling in combination on every other frame can facilitate the display of 23.98p video on a 59.94p display.

Table D-1 – Interpretation of pic_struct

Value	Indicated display of picture	Restrictions	NumClockTS
0	frame	field_pic_flag shall be 0	1
1	top field	field_pic_flag shall be 1, bottom_field_flag shall be 0	1
2	bottom field	field_pic_flag shall be 1, bottom_field_flag shall be 1	1
3	top field, bottom field, in that order	field_pic_flag shall be 0	2
4	bottom field, top field, in that order	field_pic_flag shall be 0	2
5	top field, bottom field, top field repeated, in that order	field_pic_flag shall be 0	3
6	bottom field, top field, bottom field repeated, in that order	field_pic_flag shall be 0	3
7	frame doubling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	2
8	frame tripling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	3
9..15	reserved		

NumClockTS is determined by pic_struct as specified in Table D-1. There are up to NumClockTS sets of clock timestamp information for a picture, as specified by clock_timestamp_flag[i] for each set. The sets of clock timestamp information apply to the field(s) or the frame(s) associated with the picture by pic_struct.

The contents of the clock timestamp syntax elements indicate a time of origin, capture, or alternative ideal display. This indicated time is computed as

$$\text{clockTimestamp} = ((\text{hH} * 60 + \text{mM}) * 60 + \text{sS}) * \text{time\_scale} + \text{nFrames} * (\text{num\_units\_in\_tick} * (1 + \text{nuit\_field\_based\_flag})) + \text{tOffset}, \tag{D-1}$$

in units of clock ticks of a clock with clock frequency equal to time_scale Hz, relative to some unspecified point in time for which clockTimestamp is equal to 0. Output order and DPB output timing are not affected by the value of clockTimestamp. When two or more frames with pic_struct equal to 0 are consecutive in output order and have equal values of clockTimestamp, the indication is that the frames represent the same content and that the last such frame in output order is the preferred representation.

NOTE – clockTimestamp time indications may aid display on devices with refresh rates other than those well-matched to DPB output times.

clock_timestamp_flag[i **]** equal to 1 indicates that a number of clock timestamp syntax elements are present and follow immediately. clock_timestamp_flag[i] equal to 0 indicates that the associated clock timestamp syntax elements are not present. When NumClockTS is greater than 1 and clock_timestamp_flag[i] is equal to 1 for more than one value of i, the value of clockTimestamp shall be non-decreasing with increasing value of i.

ct_type indicates the scan type (interlaced or progressive) of the source material as follows:

Two fields of a coded frame may have different values of ct_type.

When clockTimestamp is equal for two fields of opposite parity that are consecutive in output order, both with ct_type equal to 0 (progressive) or ct_type equal to 2 (unknown), the two fields are indicated to have come from the same original progressive frame. Two consecutive fields in output order shall have different values of clockTimestamp if the value of ct_type for either field is 1 (interlaced).

Table D-2 – Mapping of ct_type to source picture scan

Value	Original picture scan
0	progressive
1	interlaced
2	unknown
3	reserved

nuit_field_based_flag: Used in calculating clockTimestamp, as specified in Equation D-1.

counting_type: Specifies the method of dropping values of the n_frames as specified in Table D-3.

Table D-3 – Definition of counting_type values

Value	Interpretation
0	no dropping of n_frames count values and no use of time_offset
1	no dropping of n_frames count values
2	dropping of individual zero values of n_frames count
3	dropping of individual MaxFPS-1 values of n_frames count
4	dropping of the two lowest (value 0 and 1) n_frames counts when seconds_value is equal to 0 and minutes_value is not an integer multiple of 10
5	dropping of unspecified individual n_frames count values
6	dropping of unspecified numbers of unspecified n_frames count values
7..31	reserved

full_timestamp_flag equal to 1 specifies that the n_frames syntax element is followed by seconds_value, minutes_value, and hours_value. full_timestamp_flag equal to 0 specifies that the n_frames syntax element is followed by seconds_flag.

discontinuity_flag equal to 0 indicates that the difference between the current value of clockTimestamp and the value of clockTimestamp computed from the previous clock timestamp in output order can be interpreted the time difference between the times of origin or capture of the associated frames or fields. discontinuity_flag equal to 1 indicates that the difference between the current value of clockTimestamp and the value of clockTimestamp computed from the previous clock timestamp in output order should not be interpreted the time difference between the times of origin or capture of the associated frames or fields. When discontinuity_flag is equal to 0, the value of clockTimestamp shall be greater than or equal to all values of clockTimestamp present for the preceding picture in DPB output order.

cnt_dropped_flag specifies the skipping of one or more values of n_frames using the counting method specified by counting_type.

n_frames specifies the value of nFrames used to compute clockTimestamp. n_frames shall be less than

$$\text{MaxFPS} = \text{Ceil}(\ \text{time\_scale} \div \text{num\_units\_in\_tick}\ ) \quad \text{(D- 2)}$$

NOTE – n_frames is a frame-based counter. For field-specific timing indications, time_offset should be used to indicate a distinct clockTimestamp for each field.

When counting_type is equal to 2 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 1 and the value of n_frames for the previous picture in output order shall not be equal to 0 unless discontinuity_flag is equal to 1.

NOTE – When counting_type is equal to 2, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 12.5 frames per second with time_scale equal to 25 and num_units_in_tick equal to 2 and nuit_field_based_flag equal to 0) can be avoided by occasionally skipping over the value n_frames equal to 0 when counting (e.g., counting n_frames from 0 to 12, then incrementing seconds_value and counting n_frames from 1 to 12, then incrementing seconds_value and counting n_frames from 0 to 12, etc.).

When counting_type is equal to 3 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 0 and the value of n_frames for the previous picture in output order shall not be equal to MaxFPS – 1 unless discontinuity_flag is equal to 1.

NOTE – When counting_type is equal to 3, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 12.5 frames per second with time_scale equal to 25 and num_units_in_tick equal to 2 and nuit_field_based_flag equal to 0) can be avoided by occasionally skipping over the value n_frames equal to MaxFPS when counting (e.g., counting n_frames from 0 to 12, then incrementing seconds_value and counting n_frames from 0 to 11, then incrementing seconds_value and counting n_frames from 0 to 12, etc.).

When counting_type is equal to 4 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 2 and the specified value of sS shall be zero and the specified value of mM shall not be an integer multiple of ten and n_frames for the previous picture in output order shall not be equal to 0 or 1 unless discontinuity_flag is equal to 1.

NOTE – When counting_type is equal to 4, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 30000÷1001 frames per second with time_scale equal to 60000 and num_units_in_tick equal to 1 001 and nuit_field_based_flag equal to 1) can be reduced by occasionally skipping over the value n_frames equal to MaxFPS when counting (e.g., counting n_frames from 0 to 29, then incrementing seconds_value and counting n_frames from 0 to 29, etc., until the seconds_value is zero and minutes_value is not an integer multiple of ten, then counting n_frames from 2 to 29, then incrementing seconds_value and counting n_frames from 0 to 29, etc.). This counting method is well known in industry and is often referred to as "NTSC drop-frame" counting.

When counting_type is equal to 5 or 6 and cnt_dropped_flag is equal to 1, n_frames shall not be equal to 1 plus the value of n_frames for the previous picture in output order modulo MaxFPS unless discontinuity_flag is equal to 1.

NOTE – When counting_type is equal to 5 or 6, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates can be avoided by occasionally skipping over some values of n_frames when counting. The specific values of n_frames that are skipped are not specified when counting_type is equal to 5 or 6.

seconds_flag equal to 1 specifies that seconds_value and minutes_flag are present when full_timestamp_flag is equal to 0. seconds_flag equal to 0 specifies that seconds_value and minutes_flag are not present.

seconds_value specifies the value of sS used to compute clockTimestamp. The value of seconds_value shall be in the range of 0 to 59, inclusive. When seconds_value is not present, the previous seconds_value in decoding order shall be used as sS to compute clockTimestamp.

minutes_flag equal to 1 specifies that minutes_value and hours_flag are present when full_timestamp_flag is equal to 0 and seconds_flag is equal to 1. minutes_flag equal to 0 specifies that minutes_value and hours_flag are not present.

minutes_value specifies the value of mM used to compute clockTimestamp. The value of minutes_value shall be in the range of 0 to 59, inclusive. When minutes_value is not present, the previous minutes_value in decoding order shall be used as mM to compute clockTimestamp.

hours_flag equal to 1 specifies that hours_value is present when full_timestamp_flag is equal to 0 and seconds_flag is equal to 1 and minutes_flag is equal to 1.

hours_value specifies the value of hH used to compute clockTimestamp. The value of hours_value shall be in the range of 0 to 23, inclusive. When hours_value is not present, the previous hours_value in decoding order shall be used as hH to compute clockTimestamp.

time_offset specifies the value of tOffset used to compute clockTimestamp. The number of bits used to represent time_offset shall be equal to time_offset_length. When time_offset is not present, the value 0 shall be used as tOffset to compute clockTimestamp.

D.2.3 Pan-scan rectangle SEI message semantics

The pan-scan rectangle SEI message syntax elements specify the coordinates of a rectangle relative to the cropping rectangle of the sequence parameter set. Each coordinate of this rectangle is specified in units of one-sixteenth sample spacing relative to the luma sampling grid.

pan_scan_rect_id contains an identifying number that may be used to identify the purpose of the pan-scan rectangle (for example, to identify the rectangle as the area to be shown on a particular display device or as the area that contains a particular actor in the scene). The value of pan_scan_rect_id shall be in the range of 0 to $2^{32} - 1$, inclusive.

Values of pan_scan_rect_id from 0 to 255 and from 512 to 2^{31}-1 may be used as determined by the application. Values of pan_scan_rect_id from 256 to 511 and from 2^{31} to 2^{32}-1 are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of pan_scan_rect_id in the range of 256 to 511 or in the range of 2^{31} to 2^{32} - 1 shall ignore (remove from the bitstream and discard) it.

pan_scan_rect_cancel_flag equal to 1 indicates that the SEI message cancels the persistence of a previous pan-scan rectangle SEI message. pan_scan_rect_cancel_flag equal to 0 indicates that the SEI message does not cancel the persistence of a previous pan-scan rectangle SEI message and that pan-scan rectangle information follows.

pan_scan_cnt_minus1 specifies the number of pan-scan rectangles that are present in the SEI message. pan_scan_cnt_minus1 shall be in the range of 0 to 2, inclusive. pan_scan_cnt_minus1 equal to 0 indicates that a single pan-scan rectangle is present that applies to all fields of the decoded picture. pan_scan_cnt_minus1 shall be equal to 0 if the current picture is a field. pan_scan_cnt_minus1 equal to 1 indicates that two pan-scan rectangles are present, the first of which applies to the first field of the picture in output order and the second of which applies to the second field of the picture in output order. pan_scan_cnt_minus1 equal to 2 indicates that three pan-scan rectangles are present, the first of which applies to the first field of the picture in output order, the second of which applies to the second field of the picture in output order, and the third of which applies to a repetition of the first field as a third field in output order.

pan_scan_rect_left_offset[i **]**, **pan_scan_rect_right_offset[** i **]**, **pan_scan_rect_top_offset[** i **]**, and **pan_scan_rect_bottom_offset[** i **]**, specify, as signed integer quantities in units of one-sixteenth sample spacing relative to the luma sampling grid, the location of the pan-scan rectangle. The values of each of these four syntax elements shall be in the range of -2^{31} to 2^{31} - 1, inclusive.

The pan-scan rectangle is specified, in units of one-sixteenth sample spacing relative to a luma frame sampling grid, as the area of the rectangle with coordinates as follows:

– If frame_mbs_only_flag is equal to 1, the pan-scan rectangle has luma frame horizontal coordinates from 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] to 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1 and with vertical coordinates from 32 * frame_crop_top_offset + pan_scan_rect_top_offset[i] to 32 * (8 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1, inclusive. In this case, the value of 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] shall be less than or equal to 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1; and the value of 32 * frame_crop_top_offset + pan_scan_rect_top_offset[i] shall be less than or equal to 32 * (8 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1.
– Otherwise (frame_mbs_only_flag is equal to 0), the pan-scan rectangle has luma frame horizontal coordinates from 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] to 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1 and with vertical coordinates from 64 * frame_crop_top_offset + pan_scan_rect_top_offset[i] to 64 * (4 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1, inclusive. In this case, the value of 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] shall be less than or equal to 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1; and the value of 64 * frame_crop_top_offset + pan_scan_rect_top_offset[i] shall be less than or equal to 64 * (4 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1.

When the pan-scan rectangular area includes samples outside of the cropping rectangle, the region outside of the cropping rectangle may be filled with synthesized content (such as black video content or neutral grey video content) for display.

pan_scan_rect_repetition_period indicates whether another pan-scan rectangle SEI message with the same value of pan_scan_rect_id shall be present in the bitstream and specifies the picture order count interval within which it will be present. The value of pan_scan_rect_repetition_period shall be in the range of 0 to 16 384, inclusive. When pan_scan_cnt_minus1 is greater than 0, pan_scan_rect_repetition_period shall not be greater than 1.

pan_scan_rect_repetition_period equal to 0 specifies that the pan-scan rectangle information applies to the current decoded picture only.

pan_scan_rect_repetition_period equal to 1 specifies that the pan-scan rectangle information persists in output order until any of the following conditions are true.

– A new coded video sequence begins
– A picture in an access unit containing a pan-scan rectangle SEI message with the same value of pan_scan_rect_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

pan_scan_rect_repetition_period equal to 0 or equal to 1 indicates that another pan-scan rectangle SEI message with the same value of pan_scan_rect_id may or may not be present.

pan_scan_rect_repetition_period greater than 1 specifies that the pan-scan rectangle information persists until any of the following conditions are true.

– A new coded video sequence begins
– A picture in an access unit containing a pan-scan rectangle SEI message with the same value of pan_scan_rect_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period.

pan_scan_rect_repetition_period greater than 1 indicates that another pan-scan rectangle SEI message with the same value of pan_scan_rect_id shall be present for a picture in an access unit that is output having PicOrderCnt( ) less than or equal to PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period; unless a new coded video sequence begins without output of such a picture.

D.2.4 Filler payload SEI message semantics

This message contains a series of payloadSize bytes of value 0xFF, which can be discarded.

ff_byte shall be a byte having the value 0xFF.

D.2.5 User data registered by ITU-T Recommendation T.35 SEI message semantics

This message contains user data registered as specified by ITU-T Recommendation T.35, the contents of which are not specified by this Recommendation | International Standard.

itu_t_t35_country_code shall be a byte having a value specified as a country code by ITU-T Recommendation T.35 Annex A.

itu_t_t35_country_code_extension_byte shall be a byte having a value specified as a country code by ITU-T Recommendation T.35 Annex B.

itu_t_t35_payload_byte shall be a byte containing data registered as specified by ITU-T Recommendation T.35.

The ITU-T T.35 terminal provider code and terminal provider oriented code shall be contained in the first one or more bytes of the itu_t_t35_payload_byte, in the format specified by the Administration that issued the terminal provider code. Any remaining itu_t_t35_payload_byte data shall be data having syntax and semantics as specified by the entity identified by the ITU-T T.35 country code and terminal provider code.

D.2.6 User data unregistered SEI message semantics

This message contains unregistered user data identified by a UUID, the contents of which are not specified by this Recommendation | International Standard.

uuid_iso_iec_11578 shall have a value specified as a UUID according to the procedures of ISO/IEC 11578:1996 Annex A.

user_data_payload_byte shall be a byte containing data having syntax and semantics as specified by the UUID generator.

D.2.7 Recovery point SEI message semantics

The recovery point SEI message assists a decoder in determining when the decoding process will produce acceptable pictures for display after the decoder initiates random access or after the encoder indicates a broken link in the sequence. When the decoding process is started with the access unit in decoding order associated with the recovery point SEI message, all decoded pictures at or subsequent to the recovery point in output order specified in this SEI message are indicated to be correct or approximately correct in content. Decoded pictures produced by random access at or before the picture associated with the recovery point SEI message need not be correct in content until the indicated recovery point, and the operation of the decoding process starting at the picture associated with the recovery point SEI message may contain references to pictures not available in the decoded picture buffer.

In addition, by use of the broken_link_flag, the recovery point SEI message can indicate to the decoder the location of some pictures in the bitstream that can result in serious visual artefacts if displayed, even when the decoding process was begun at the location of a previous IDR access unit in decoding order.

NOTE – The broken_link_flag can be used by encoders to indicate the location of a point after which the decoding process for the decoding of some pictures may cause references to pictures that, though available for use in the decoding process, are not the pictures that were used for reference when the bitstream was originally encoded (e.g., due to a splicing operation performed during the generation of the bitstream).

The recovery point is specified as a count in units of access units subsequent to the current access unit at the position of the SEI message.

NOTE – When HRD information is present in the bitstream, a buffering period SEI message should be associated with the access unit associated with the recovery point SEI message in order to establish initialisation of the HRD buffer model after a random access.

recovery_frame_cnt specifies the recovery point of output pictures in output order. All decoded pictures in output order are indicated to be correct or approximately correct in content starting at the output order position of the reference picture having the frame_num equal to the frame_num of the VCL NAL units for the current access unit incremented by recovery_frame_cnt in modulo MaxFrameNum arithmetic. recovery_frame_cnt shall be in the range of 0 to MaxFrameNum – 1, inclusive.

exact_match_flag indicates whether decoded pictures at and subsequent to the specified recovery point in output order derived by starting the decoding process at the access unit associated with the recovery point SEI message shall be an exact match to the pictures that would be produced by starting the decoding process at the location of a previous IDR access unit in the NAL unit stream. The value 0 indicates that the match need not be exact and the value 1 indicates that the match shall be exact.

When decoding starts from the location of the recovery point SEI message, all references to not available reference pictures shall be inferred as references to pictures containing only intra macroblocks and having sample values given by Y samples equal to 128, Cb samples equal to 128, and Cr samples equal to 128 (mid-level grey) for purposes of determining the conformance of the value of exact_match_flag.

NOTE – When performing random access, decoders should infer all references to not available reference pictures as references to pictures containing only intra macroblocks and having sample values given by Y equal to 128, Cb equal to 128, and Cr equal to 128 (mid-level grey), regardless of the value of exact_match_flag.

When exact_match_flag is equal to 0, the quality of the approximation at the recovery point is chosen by the encoding process and is not specified by this Recommendation | International Standard.

broken_link_flag indicates the presence or absence of a broken link in the NAL unit stream at the location of the recovery point SEI message.

- If broken_link_flag is equal to 1, pictures produced by starting the decoding process at the location of a previous IDR access unit may contain undesirable visual artefacts to the extent that decoded pictures at and subsequent to the access unit associated with the recovery point SEI message in decoding order should not be displayed until the specified recovery point in output order.

- Otherwise (broken_link_flag is equal to 0), no indication is given regarding any potential presence of visual artefacts.

Regardless of the value of the broken_link_flag, pictures subsequent to the specified recovery point in output order are specified to be correct or approximately correct in content.

NOTE – When a sub-sequence information SEI message is present in conjunction with a recovery point SEI message in which broken_link_flag is equal to 1 and when sub_seq_layer_num is equal to 0, sub_seq_id should be different from the latest sub_seq_id for sub_seq_layer_num equal to 0 that was decoded prior to the location of the recovery point SEI message. When broken_link_flag is equal to 0, the sub_seq_id in sub-sequence layer 0 should remain unchanged.

changing_slice_group_idc equal to 0 indicates that decoded pictures are correct or approximately correct in content at and subsequent to the recovery point in output order if all macroblocks of the primary coded pictures are decoded within the changing slice group period, i.e., the period between the access unit associated with the recovery point SEI message (inclusive) and the specified recovery point (exclusive) in decoding order. changing_slice_group_idc shall be equal to 0 if num_slice_groups_minus1 is equal to 0 in any primary coded picture within the changing slice group period.

When changing_slice_group_idc is equal to 1 or 2, num_slice_groups_minus1 shall be equal to 1 and the macroblock-to-slice-group map type 3, 4, or 5 shall be applied in each primary coded picture in the changing slice group period.

changing_slice_group_idc equal to 1 indicates that within the changing slice group period no sample values outside the decoded macroblocks covered by slice group 0 are used for inter prediction of any macroblock within slice group 0. In addition, changing_slice_group_idc equal to 1 indicates that when all macroblocks in slice group 0 within the changing slice group period are decoded, decoded pictures will be correct or approximately correct in content at and subsequent to the specified recovery point in output order regardless of whether any macroblocks in slice group 1 within the changing slice group period are decoded.

changing_slice_group_idc equal to 2 indicates that within the changing slice group period no sample values outside the decoded macroblocks covered by slice group 1 are used for inter prediction of any macroblock within slice group 1. In addition, changing_slice_group_idc equal to 2 indicates that when all macroblocks in slice group 1 within the changing slice group period are decoded, decoded pictures will be correct or approximately correct in content at and subsequent to the specified recovery point in output order regardless of whether any macroblock in slice group 0 within the changing slice group period are decoded.

changing_slice_group_idc shall be in the range of 0 to 2, inclusive.

D.2.8 Decoded reference picture marking repetition SEI message semantics

The decoded reference picture marking repetition SEI message is used to repeat the decoded reference picture marking syntax structure that was located in the slice header of an earlier picture in the sequence in decoding order.

original_idr_flag shall be equal to 1 if the decoded reference picture marking syntax structure occurred originally in an IDR picture. original_idr_flag shall be equal to 0 if the repeated decoded reference picture marking syntax structure did not occur in an IDR picture originally.

original_frame_num shall be equal to the frame_num of the picture where the repeated decoded reference picture marking syntax structure originally occurred.

original_field_pic_flag shall be equal to the field_pic_flag of the picture where the repeated decoded reference picture marking syntax structure originally occurred.

original_bottom_field_flag shall be equal to the bottom_field_flag of the picture where the repeated decoded reference picture marking syntax structure originally occurred.

dec_ref_pic_marking( ) shall contain a copy of the decoded reference picture marking syntax structure of the picture whose frame_num was original_frame_num. The nal_unit_type used for specification of the repeated dec_ref_pic_marking( ) syntax structure shall be the nal_unit_type of the slice header(s) of the picture whose frame_num was original_frame_num (i.e., nal_unit_type as used in subclause 7.3.3.3 shall be considered equal to 5 if original_idr_flag is equal to 1 and shall not be considered equal to 5 if original_idr_flag is equal to 0).

D.2.9 Spare picture SEI message semantics

This SEI message indicates that certain slice group map units, called spare slice group map units, in one or more decoded reference pictures resemble the co-located slice group map units in a specified decoded picture called the target picture. A spare slice group map unit may be used to replace a co-located, incorrectly decoded slice group map unit, in the target picture. A decoded picture containing spare slice group map units is called a spare picture.

For all spare pictures identified in a spare picture SEI message, the value of frame_mbs_only_flag shall be equal to the value of frame_mbs_only_flag of the target picture in the same SEI message.

- If the target picture is a decoded field, all spare pictures identified in the same SEI message shall be decoded fields.
- Otherwise (the target picture is a decoded frame), all spare pictures identified in the same SEI message shall be decoded frames.

For all spare pictures identified in a spare picture SEI message, the values of pic_width_in_mbs_minus1 and pic_height_in_map_units_minus1 shall be equal to the values of pic_width_in_mbs_minus1 and pic_height_in_map_units_minus1, respectively, of the target picture in the same SEI message. The picture associated (as specified in subclause 7.4.1.2.3) with this message shall appear after the target picture, in decoding order.

target_frame_num indicates the frame_num of the target picture.

spare_field_flag equal to 0 indicates that the target picture and the spare pictures are decoded frames. spare_field_flag equal to 1 indicates that the target picture and the spare pictures are decoded fields.

target_bottom_field_flag equal to 0 indicates that the target picture is a top field. target_bottom_field_flag equal to 1 indicates that the target picture is a bottom field.

A target picture is a decoded reference picture whose corresponding primary coded picture precedes the current picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to target_frame_num, spare_field_flag and target_bottom_field_flag, respectively.

num_spare_pics_minus1 indicates the number of spare pictures for the specified target picture. The number of spare pictures is equal to num_spare_pics_minus1 + 1. The value of num_spare_pics_minus1 shall be in the range of 0 to 15, inclusive.

delta_spare_frame_num[i] is used to identify the spare picture that contains the i-th set of spare slice group map units, hereafter called the i-th spare picture, as specified below. The value of delta_spare_frame_num[i] shall be in the range of 0 to MaxFrameNum - 1 - !spare_field_flag, inclusive.

The frame_num of the i-th spare picture, spareFrameNum[i], is derived as follows for all values of i from 0 to num_spare_pics_minus1, inclusive:

```
candidateSpareFrameNum = target_frame_num - !spare_field_flag
for ( i = 0; i <= num_spare_pics_minus1; i++ ) {
    if( candidateSpareFrameNum < 0 )
        candidateSpareFrameNum = MaxFrameNum – 1
```

```
        spareFrameNum[ i ] = candidateSpareFrameNum – delta_spare_frame_num[ i ]        (D-3)
        if( spareFrameNum[ i ] < 0 )
            spareFrameNum[ i ] = MaxFrameNum + spareFrameNum[ i ]
        candidateSpareFrameNum = spareFrameNum[ i ] - !spare_field_flag
    }
```

spare_bottom_field_flag[i] equal to 0 indicates that the i-th spare picture is a top field. spare_bottom_field_flag[i] equal to 1 indicates that the i-th spare picture is a bottom field.

The 0-th spare picture is a decoded reference picture whose corresponding primary coded picture precedes the target picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to spareFrameNum[0], spare_field_flag and spare_bottom_field_flag[0], respectively. The i-th spare picture is a decoded reference picture whose corresponding primary coded picture precedes the (i - 1)-th spare picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to spareFrameNum[i], spare_field_flag and spare_bottom_field_flag[i], respectively.

spare_area_idc[i] indicates the method used to identify the spare slice group map units in the i-th spare picture. spare_area_idc[i] shall be in the range of 0 to 2, inclusive. spare_area_idc[i] equal to 0 indicates that all slice group map units in the i-th spare picture are spare units. spare_area_idc[i] equal to 1 indicates that the value of the syntax element spare_unit_flag[i][j] is used to identify the spare slice group map units. spare_area_idc[i] equal to 2 indicates that the zero_run_length[i][j] syntax element is used to derive the values of spareUnitFlagInBoxOutOrder[i][j], as described below.

spare_unit_flag[i][j] equal to 0 indicates that the j-th slice group map unit in raster scan order in the i-th spare picture is a spare unit. spare_unit_flag[i][j] equal to 1 indicates that the j-th slice group map unit in raster scan order in the i-th spare picture is not a spare unit.

zero_run_length[i][j] is used to derive the values of spareUnitFlagInBoxOutOrder[i][j] when spare_area_idc[i] is equal to 2. In this case, the spare slice group map units identified in spareUnitFlagInBoxOutOrder[i][j] appear in counter-clockwise box-out order, as specified in subclause 8.2.2.4, for each spare picture. spareUnitFlagInBoxOutOrder[i][j] equal to 0 indicates that the j-th slice group map unit in counter-clockwise box-out order in the i-th spare picture is a spare unit. spareUnitFlagInBoxOutOrder[i][j] equal to 1 indicates that the j-th slice group map unit in counter-clockwise box-out order in the i-th spare picture is not a spare unit.

When spare_area_idc[0] is equal to 2, spareUnitFlagInBoxOutOrder[0][j] is derived as follows:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ 0 ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 0                                      (D-4)
    spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 1
}
```

When spare_area_idc[i] is equal to 2 and the value of i is greater than 0, spareUnitFlagInBoxOutOrder[i][j] is derived as follows:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ i ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ i ][ j ] = spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]    (D-5)
    spareUnitFlagInBoxOutOrder[ i ][ j ] = !spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]
}
```

D.2.10 Scene information SEI message semantics

A scene and a scene transition are herein defined as a set of consecutive pictures in output order.

NOTE - Decoded pictures within one scene generally have similar content. The scene information SEI message is used to label pictures with scene identifiers and to indicate scene changes. The message specifies how the source pictures for the labelled pictures were created. The decoder may use the information to select an appropriate algorithm to conceal transmission errors. For example, a specific algorithm may be used to conceal transmission errors that occurred in pictures belonging to a gradual scene transition. Furthermore, the scene information SEI message may be used in a manner determined by the application, such as for indexing the scenes of a coded sequence.

A scene information SEI message labels all pictures, in decoding order, from the primary coded picture to which the SEI message is associated (inclusive), as specified in subclause 7.4.1.2.3, to the primary coded picture to which the next scene information SEI message, in decoding order, is associated (exclusive). These pictures are herein referred to as the target pictures.

scene_info_present_flag equal to 0 indicates that the scene or scene transition to which the target pictures belong is unspecified. scene_info_present_flag equal to 1 indicates that the target pictures belong to the same scene or scene transition.

scene_id identifies the scene to which the target pictures belong. The value of scene_id shall be the same as the value of the scene_id of the previous picture, in output order, that is marked with a value of scene_transition_type less than 4, if the value of scene_transition_type of the target pictures is greater than 3 and if the target pictures and the previous picture (in output order) that is marked with a value of scene_transition_type less than 4 originate from the same source scene.

The value of scene_id shall be in the range of 0 to 2^{32}-1, inclusive. Values of scene_id in the range of 0 to 255, inclusive, and in the range of 512 to $2^{31} – 1$, inclusive, may be used as determined by the application. Values of scene_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to $2^{32} – 1$, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of scene_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

scene_transition_type specifies in which type of a scene transition (if any) the target pictures are involved. The valid values of scene_transition_type are specified in Table D-4.

Table D-4 – scene_transition_type values.

Value	Description
0	No transition
1	Fade to black
2	Fade from black
3	Unspecified transition from or to constant colour
4	Dissolve
5	Wipe
6	Unspecified mixture of two scenes

When scene_transition_type is greater than 3, the target pictures include contents both from the scene labelled by its scene_id and the next scene, in output order, which is labelled by second_scene_id (see below). The term “the current scene” is used to indicate the scene labelled by scene_id. The term “the next scene” is used to indicate the scene labelled by second_scene_id. It is not required for any following picture, in output order, to be labelled with scene_id equal to second_scene_id of the current SEI message.

Scene transition types are specified as follows.

“No transition” specifies that the target pictures are not involved in a gradual scene transition.

NOTE - When two consecutive pictures in output order have scene_transition_type equal to 0 and different values of scene_id, a scene cut occurred between the two pictures.

“Fade to black” indicates that the target pictures are part of a sequence of pictures, in output order, involved in a fade to black scene transition, i.e., the luma samples of the scene gradually approach zero and the chroma samples of the scene gradually approach 128.

NOTE – When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Fade to black", the later one, in output order, is darker than the previous one.

“Fade from black” indicates that the target pictures are part of a sequence of pictures, in output order, involved in a fade from black scene transition, i.e., the luma samples of the scene gradually diverge from zero and the chroma samples of the scene may gradually diverge from 128.

NOTE – When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Fade from black", the later one in output order is lighter than the previous one.

“Dissolve” indicates that the sample values of each target picture (before encoding) were generated by calculating a sum of co-located weighted sample values of a picture from the current scene and a picture from the next scene. The weight of the current scene gradually decreases from full level to zero level, whereas the weight of the next scene gradually increases from zero level to full level. When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Dissolve", the weight of the current scene for the later one, in output order, is less than the weight of the current scene for the previous one, and the weight of the next scene for the later one, in output order, is larger than the weight of the next scene for the previous one.

“Wipe” indicates that some of the sample values of each target picture (before encoding) were generated by copying co-located sample values of a picture in the next scene. When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Wipe", the number of samples copied from the next scene for the later one in output order is larger than the previous one.

second_scene_id identifies another scene in the gradual scene transition in which the target pictures are involved. The value of second_scene_id shall be the same as the value of scene_id of the next picture, in output order, that is marked with a value of scene_transition_type less than 4, if the target pictures and the next picture (in output order) that is marked with a value of scene_transition_type less than 4 originate from the same source scene. The value of second_scene_id shall not be equal to the value of scene_id in the previous picture, in output order.

If the value of scene_id of a picture is equal to the value of scene_id of the following picture, in output order, and the value of scene_transition_type in both of these pictures is equal to 0, these two pictures belong to the same scene. If the value of scene_id of a picture is equal to the value of scene_id of the following picture, in output order, and the value of scene_transition_type in both of these pictures is less than 4, these two pictures originate from the same source scene. If the values of scene_id, scene_transition_type and second_scene_id of a picture are equal to the values of scene_id, scene_transition_type and second_scene_id (respectively) of the following picture, in output order, these two pictures belong to the same scene transition.

The value of second_scene_id shall be in the range of 0 to 2^{32}-1, inclusive. Values of second_scene_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31}-1, inclusive, may be used as determined by the application. Values of second_scene_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of second_scene_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32}-1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.11 Sub-sequence information SEI message semantics

The sub-sequence information SEI message is used to indicate the position of a picture in data dependency hierarchy that consists of sub-sequence layers and sub-sequences.

A sub-sequence layer contains a subset of the coded pictures in a sequence. Sub-sequence layers are numbered with non-negative integers. A layer having a larger layer number is a higher layer than a layer having a smaller layer number. The layers are ordered hierarchically based on their dependency on each other so that any picture in a layer shall not be predicted from any picture on any higher layer.

NOTE – In other words, any picture in layer 0 must not be predicted from any picture in layer 1 or above, pictures in layer 1 may be predicted from layer 0, pictures in layer 2 may be predicted from layers 0 and 1, etc.

NOTE: The subjective quality is expected to increase along with the number of decoded layers.

A sub-sequence is a set of coded pictures within a sub-sequence layer. A picture shall reside in one sub-sequence layer and in one sub-sequence only. Any picture in a sub-sequence shall not be predicted from any picture in another sub-sequence in the same or in a higher sub-sequence layer. A sub-sequence in layer 0 can be decoded independently of any picture that does not belong to the sub-sequence.

The sub-sequence information SEI message concerns the current access unit. The primary coded picture in the access unit is herein referred to as the current picture.

The sub-sequence information SEI message shall not be present unless gaps_in_frame_num_value_allowed_flag in the sequence parameter set referenced by the picture associated with the sub-sequence SEI message is equal to 1.

sub_seq_layer_num specifies the sub-sequence layer number of the current picture. When sub_seq_layer_num is greater than 0, memory management control operations shall not be used in any slice header of the current picture. When the current picture resides in a sub-sequence whose first picture in decoding order is an IDR picture, the value of sub_seq_layer_num shall be equal to 0. For a non-paired reference field, the value of sub_seq_layer_num shall be equal to 0. sub_seq_layer_num shall be in the range of 0 to 255, inclusive.

sub_seq_id identifies the sub-sequence within a layer. When the current picture resides in a sub-sequence whose first picture in decoding order is an IDR picture, the value of sub_seq_id shall be the same as the value of idr_pic_id of the IDR picture. sub_seq_id shall be in the range of 0 to 65535, inclusive.

first_ref_pic_flag equal to 1 specifies that the current picture is the first reference picture of the sub-sequence in decoding order. Otherwise, the first_ref_pic_flag shall be equal to 0.

leading_non_ref_pic_flag equal to 1 specifies that the current picture is a non-reference picture preceding any reference picture in decoding order within the sub-sequence or that the sub-sequence contains no reference pictures. Otherwise, the leading_non_ref_pic_flag shall be equal to 0.

last_pic_flag equal to 1 specifies that the current picture is the last picture of the sub-sequence (in decoding order), including all reference and non-reference pictures of the sub-sequence. For any other pictures, the last_pic_flag shall be 0.

The current picture is the first picture of a sub-sequence in decoding order, if no earlier picture in decoding order is labelled with the same sub_seq_id and sub_seq_layer_num as the current picture, or if the leading_non_ref_pic_flag is equal to 1 and the leading_non_ref_pic_flag is equal to 0 in the previous picture in decoding order having the same sub_seq_id and sub_seq_layer_num as the current picture, or if the first_ref_pic_flag is equal to 1 and the leading_non_ref_pic_flag is equal to 0 in the previous picture in decoding order having the same sub_seq_id and sub_seq_layer_num as the current picture, or if the last_pic_flag is equal to 1 in the previous picture in decoding order having the same sub_seq_id and sub_seq_layer_num as the current picture. Otherwise, the current picture belongs to the same sub-sequence as the previous picture in decoding order having the same sub_seq_id and sub_seq_layer_num as the current picture.

sub_seq_frame_num_flag equal to 0 specifies that sub_seq_frame_num is not present. sub_seq_frame_num_flag equal to 1 specifies that sub_seq_frame_num is present.

sub_seq_frame_num shall be equal to 0 for the first reference picture of the sub-sequence and for any non-reference picture preceding the first reference picture of the sub-sequence in decoding order.

- If the current picture is not the second field of a complementary field pair, sub_seq_frame_num shall be incremented by 1, in modulo MaxFrameNum operation, relative to the previous reference picture, in decoding order, that belongs to the sub-sequence.
- Otherwise (the current picture is the second field of a complementary field pair), the value of sub_seq_frame_num shall be the same as the value of sub_seq_frame_num for the first field of the complementary field pair. If a coded picture sub_seq_frame_num shall be less than MaxFrameNum.

When the current picture is an IDR picture, it shall start a new sub-sequence in sub-sequence layer 0. Thus, the sub_seq_layer_num shall be 0, the sub_seq_id shall be different from the previous sub-sequence in sub-sequence layer 0, first_ref_pic_flag shall be 1, and leading_non_ref_pic_flag shall be equal to 0.

If the sub-sequence information SEI message is present for both coded fields of a complementary field pair, the values of sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag and sub_seq_frame_num, when present, shall be identical for both of these pictures. If the sub-sequence information SEI message is present only for one coded field of a complementary field pair, the values of sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag and sub_seq_frame_num, when present, are also applicable to the other coded field of the complementary field pair.

D.2.12 Sub-sequence layer characteristics SEI message semantics

The sub-sequence layer characteristics SEI message specifies the characteristics of sub-sequence layers.

num_sub_seq_layers_minus1 plus 1 specifies the number of sub-sequence layers in the sequence. num_sub_seq_layers_minus1 shall be in the range of 0 to 255, inclusive.

A pair of average_bit_rate and average_frame_rate characterizes each sub-sequence layer. The first pair of average_bit_rate and average_frame_rate specifies the characteristics of sub-sequence layer 0. When present, the second pair specifies the characteristics of sub-sequence layers 0 and 1 jointly. Each pair in decoding order specifies the characteristics for a range of sub-sequence layers from layer number 0 to the layer number specified by the layer loop counter. The values are in effect from the point they are decoded until an update of the values is decoded.

accurate_statistics_flag equal to 1 indicates that the values of average_bit_rate and average_frame_rate are rounded from statistically correct values. accurate_statistics_flag equal to 0 indicates that the average_bit_rate and the average_frame_rate are estimates and may deviate somewhat from the correct values.

When accurate_statistics_flag is equal to 0, the quality of the approximation used in the computation of the values of average_bit_rate and the average_frame_rate is chosen by the encoding process and is not specified by this Recommendation | International Standard.

average_bit_rate gives the average bit rate in units of 1000 bits per second. All NAL units in the range of sub-sequence layers specified above are taken into account in the calculation. The average bit rate is derived according to the picture removal time specified in Annex C of the Recommendation | International Standard. In the following, bTotal is the number of bits in all NAL units succeeding a sub-sequence layer characteristics SEI message (including the bits of the NAL units of the current picture) and preceding the next sub-sequence layer characteristics SEI message or the end of the stream. t_1 is the removal time (in seconds) of the current picture, and t_2 is the removal time (in seconds) of the latest picture before the next sub-sequence layer characteristics SEI message or the end of the stream.

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled.

- If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_bit\_rate} == \text{Round}(\text{bTotal} \div ((t_2 - t_1) * 1000)) \quad \text{(D-6)}$$

- Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_bit\_rate} == 0 \quad \text{(D-7)}$$

average_frame_rate gives the average frame rate in units of frames/(256 seconds). All NAL units in the range of sub-sequence layers specified above are taken into account in the calculation. In the following, fTotal is the number of frames between the current picture (inclusive) and the next sub-sequence layer characteristics SEI message or the end of the stream. t_1 is the removal time (in seconds) of the current picture, and t_2 is the removal time (in seconds) of the latest picture before the next sub-sequence layer characteristics SEI message or the end of the stream.

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled.

- If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_frame\_rate} == \text{Round}(\text{ fTotal} * 256 \div (t_2 - t_1)) \quad \text{(D-8)}$$

- Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_frame\_rate} == 0 \quad \text{(D-9)}$$

D.2.13 Sub-sequence characteristics SEI message semantics

The sub-sequence characteristics SEI message indicates the characteristics of a sub-sequence. It also indicates inter prediction dependencies between sub-sequences.

sub_seq_layer_num specifies the sub-sequence layer number to which the sub-sequence characteristics SEI message applies. sub_seq_layer_num shall be in the range of 0 to 255, inclusive.

sub_seq_id specifies the sub-sequence within a layer to which the sub-sequence characteristics SEI message applies. sub_seq_id shall be in the range of 0 to 65535, inclusive.

This message applies to the next sub-sequence in decoding order having the specified sub_seq_layer_num and sub_seq_id. This sub-sequence is herein called the target sub-sequence.

duration_flag equal to 0 indicates that the duration of the target sub-sequence is not specified.

sub_seq_duration specifies the duration of the target sub-sequence in clock ticks of a 90-kHz clock.

average_rate_flag equal to 0 indicates that the average bit rate and the average frame rate of the target sub-sequence are unspecified.

accurate_statistics_flag indicates how reliable the values of average_bit_rate and average_frame_rate are. accurate_statistics_flag equal to 1, indicates that the average_bit_rate and the average_frame_rate are rounded from statistically correct values. accurate_statistics_flag equal to 0 indicates that the average_bit_rate and the average_frame_rate are estimates and may deviate from the statistically correct values.

average_bit_rate gives the average bit rate in (1000 bits)/second of the target sub-sequence. All NAL units of the target sub-sequence are taken into account in the calculation. The average bit rate is derived according to the picture removal time specified in subclause C.1.2. In the following, B is the number of bits in all NAL units in the sub-sequence. t_1 is the removal time (in seconds) of the first picture of the sub-sequence (in decoding order), and t_2 is the removal time (in seconds) of the last picture of the sub-sequence (in decoding order).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled.

- If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_bit\_rate} == \text{Round}(B \div ((t_2 - t_1) * 1000)) \quad \text{(D-10)}$$

- Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_bit\_rate} == 0 \quad \text{(D-11)}$$

average_frame_rate gives the average frame rate in units of frames/(256 seconds) of the target sub-sequence. All NAL units of the target sub-sequence are taken into account in the calculation. The average frame rate is derived according to the picture removal time specified in subclause C.1.2. In the following, C is the number of frames in the sub-sequence. t_1 is the removal time (in seconds) of the first picture of the sub-sequence (in decoding order), and t_2 is the removal time (in seconds) of the last picture of the sub-sequence (in decoding order).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled.

- If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_frame\_rate} == \text{Round}(C * 256 \div (t_2 - t_1)) \quad \text{(D-12)}$$

- Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_frame\_rate} == 0 \quad \text{(D-13)}$$

num_referenced_subseqs gives the number of sub-sequences that contain pictures that are used as reference pictures for inter prediction in the pictures of the target sub-sequence. num_referenced_subseqs shall be in the range of 0 to 255, inclusive.

ref_sub_seq_layer_num, ref_sub_seq_id, and **ref_sub_seq_direction** identify the sub-sequence that contains pictures that are used as reference pictures for inter prediction in the pictures of the target sub-sequence.

- If ref_sub_seq_direction is equal to 0, a set of candidate sub-sequences consists of the sub-sequences whose sub_seq_id is equal to ref_sub_seq_id, which reside in the sub-sequence layer having sub_seq_layer_num equal to ref_sub_seq_layer_num, and whose first picture in decoding order precedes the first picture of the target sub-sequence in decoding order.
- Otherwise (ref_sub_seq_direction is equal to 1), a set of candidate sub-sequences consists of the sub-sequences whose sub_seq_id is equal to ref_sub_seq_id, which reside in the sub-sequence layer having sub_seq_layer_num equal to ref_sub_seq_layer_num, and whose first picture in decoding order succeeds the first picture of the target sub-sequence in decoding order.

The sub-sequence used as a reference for the target sub-sequence is the sub-sequence among the set of candidate sub-sequences whose first picture is the closest to the first picture of the target sub-sequence in decoding order.

D.2.14 Full-frame freeze SEI message semantics

The full-frame freeze SEI message indicates that the contents of the entire prior displayed video frame in output order should be kept unchanged, without updating the display using the contents of the current decoded picture.

full_frame_freeze_repetition_period indicates whether another full-frame freeze SEI message shall be present in the bitstream and specifies the picture order count interval within which another full-frame freeze SEI message or a full-frame freeze release SEI message will be present. The value of full_frame_freeze_repetition_period shall be in the range of 0 to 16 484, inclusive.

full_frame_freeze_repetition_period equal to 0 specifies that the full-frame freeze SEI message applies to the current decoded picture only.

full_frame_freeze_repetition_period equal to 1 specifies that the full-frame freeze SEI message persists in output order until any of the following conditions are true.

– A new coded video sequence begins
– A picture in an access unit containing full-frame freeze release SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

full_frame_freeze_repetition_period greater than 1 specifies that the full-frame freeze SEI message persists until any one of the following conditions are true.

– A new coded video sequence begins
– A picture in an access unit containing a full-frame freeze release SEI message or a full-frame freeze release SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period.

full_frame_freeze_repetition_period greater than 1 indicates that another full-frame freeze SEI message or a full-frame freeze release SEI message shall be present for a picture in an access unit that is output having PicOrderCnt( ) less than or equal to PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period; unless a new coded video sequence begins without output of such a picture.

D.2.15 Full-frame freeze release SEI message semantics

The full-frame freeze release SEI message indicates that the update of the displayed video frame should resume, starting with the contents of the current decoded picture and continuing for subsequent pictures in output order. The full-frame freeze release SEI message cancels the effect of any full-frame freeze SEI message sent with pictures that precede the current picture in output order.

D.2.16 Full-frame snapshot SEI message semantics

The full-frame snapshot SEI message indicates that the current frame is labelled for use as determined by the application as a still-image snapshot of the video content.

snapshot_id specifies a snapshot identification number. snapshot_id shall be in the range of 0 to 2^{32} - 1, inclusive.

Values of snapshot_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31}-1, inclusive, may be used as determined by the application. Values of snapshot_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of snapshot_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32}-1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.17 Progressive refinement segment start SEI message semantics

The progressive refinement segment start SEI message specifies the beginning of a set of consecutive coded pictures that is labelled as the current picture followed by a sequence of one or more pictures of refinement of the quality of the current picture, rather than as a representation of a continually moving scene.

The tagged set of consecutive coded pictures shall continue until one of following conditions is true. When a condition below becomes true, the next slice to be decoded does not belong to the tagged set of consecutive coded pictures.

1. The next slice to be decoded belongs to an IDR picture.
2. num_refinement_steps_minus1 is greater than 0 and the frame_num of the next slice to be decoded is (currFrameNum + num_refinement_steps_minus1 + 1) % MaxFrameNum, where currFrameNum is the value of frame_num of the picture in the access unit containing the SEI message.
3. num_refinement_steps_minus1 is 0 and a progressive refinement segment end SEI message with the same progressive_refinement_id as the one in this SEI message is decoded.

The decoding order of picture within the tagged set of consecutive pictures should be the same as their output order. **progressive_refinement_id** specifies an identification number for the progressive refinement operation. progressive_refinement_id shall be in the range of 0 to 2^{32} - 1, inclusive.

Values of progressive_refinement_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31} - 1, inclusive, may be used as determined by the application. Values of progressive_refinement_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of progressive_refinement_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

num_refinement_steps_minus1 specifies the number of reference frames in the tagged set of consecutive coded pictures. num_refinement_steps_minus1 equal to 0 specifies that the number of reference frames in the tagged set of consecutive coded pictures is unknown. Otherwise, the number of reference frames in the tagged set of consecutive coded pictures is equal to num_refinement_steps_minus1 + 1. num_refinement_steps_minus1 shall be in the range of 0 to MaxFrameNum - 1, inclusive.

D.2.18 Progressive refinement segment end SEI message semantics

The progressive refinement segment end SEI message specifies the end of a set of consecutive coded pictures that has been labelled by use of a progressive refinement segment start SEI message as an initial picture followed by a sequence of one or more pictures of the refinement of the quality of the initial picture, and ending with the current picture.

progressive_refinement_id specifies an identification number for the progressive refinement operation. progressive_refinement_id shall be in the range of 0 to 2^{32} - 1, inclusive.

The progressive refinement segment end SEI message specifies the end of any progressive refinement segment previously started using a progressive refinement segment start SEI message with the same value of progressive_refinement_id.

Values of progressive_refinement_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31} - 1, inclusive, may be used as determined by the application. Values of progressive_refinement_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32} - 1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of progressive_refinement_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.19 Motion-constrained slice group set SEI message semantics

This SEI message indicates that inter prediction over slice group boundaries is constrained as specified below. When present, the message shall only appear where it is associated, as specified in subclause 7.4.1.2.3, with an IDR access unit.

The target picture set for this SEI message contains all consecutive primary coded pictures in decoding order starting with the associated primary coded IDR picture (inclusive) and ending with the following primary coded IDR picture (exclusive) or with the very last primary coded picture in the bitstream (inclusive) in decoding order if there is no following primary coded IDR picture. The slice group set is a collection of one or more slice groups, identified by the slice_group_id[i] syntax element.

This SEI message indicates that, for each picture in the target picture set, the inter prediction process is constrained as follows: No sample value outside the slice group set, and no sample value at a fractional sample position that is derived using one or more sample values outside the slice group set is used to inter predict any sample within the slice group set.

num_slice_groups_in_set_minus1 + 1 specifies the number of slice groups in the slice group set. The allowed range of num_slice_groups_in_set_minus1 is 0 to num_slice_groups_minus1, inclusive. The allowed range of num_slice_groups_minus1 is specified in Annex A.

slice_group_id[i **]** identifies the slice group(s) contained within the slice group set. The allowed range is from 0 to num_slice_groups_in_set_minus1, inclusive. The size of the slice_group_id[i] syntax element is Ceil(Log2(num_slice_groups_minus1 + 1)) bits.

exact_sample_value_match_flag equal to 0 indicates that, within the target picture set, when the macroblocks that do not belong to the slice group set are not decoded, the value of each sample in the slice group set need not be exactly the same as the value of the same sample when all the macroblocks are decoded. exact_sample_value_match_flag equal to 1 indicates that, within the target picture set, when the macroblocks that do not belong to the slice group set are not decoded, the value of each sample in the slice group set shall be exactly the same as the value of the same sample when all the macroblocks in the target picture set are decoded.

Note - When disable_deblocking_filter_idc is equal to 2 in all slices in the target picture set, exact_sample_value_match_flag should be 1.

pan_scan_rect_flag equal to 0 specifies that pan_scan_rect_id is not present. pan_scan_rect_flag equal to 1 specifies that pan_scan_rect_id is present.

pan_scan_rect_id indicates that the specified slice group set covers at least the pan-scan rectangle identified by pan_scan_rect_id within the target picture set.

Note - Multiple motion_constrained_slice_group_set SEI messages may be associated with the same IDR picture. Consequently, more than one slice group set may be active within a target picture set.

Note - The size, shape, and location of the slice groups in the slice group set may change within the target picture set.

D.2.20 Reserved SEI message semantics

This message consists of data reserved for future backward-compatible use by ITU-T | ISO/IEC. Encoders conforming to this Recommendation | International Standard shall not send reserved SEI messages until and unless the use of such messages has been specified by ITU-T | ISO/IEC. Decoders conforming to this Recommendation | International Standard that encounter reserved SEI messages shall discard their content without effect on the decoding process, except as specified in future Recommendations | International Standards specified by ITU-T | ISO/IEC. reserved_sei_message_payload_byte is a byte reserved for future use by ITU-T | ISO/IEC.

Annex E
Video usability information

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

This Annex specifies syntax and semantics of the VUI parameters of the sequence parameter sets.

VUI parameters are not required for constructing the luma or chroma samples by the decoding process. Conforming decoders are not required to process this information for output order conformance to this Recommendation | International Standard (see Annex C for the specification of conformance). Some VUI parameters are required to check bitstream conformance and for output timing decoder conformance.

In Annex E, specification for presence of VUI parameters is also satisfied when those parameters (or some subset of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. When present in the bitstream, VUI parameters shall follow the syntax and semantics specified in subclauses 7.3.2.1 and 7.4.2.1 and this annex. When the content of VUI parameters is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the VUI parameters is not required to use the same syntax specified in this annex. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

E.1 VUI syntax

E.1.1 VUI parameters syntax

vui_parameters( ) {	C	Descriptor
aspect_ratio_info_present_flag	0	u(1)
if(aspect_ratio_info_present_flag) {		
aspect_ratio_idc	0	u(8)
if(aspect_ratio_idc = = Extended_SAR) {		
sar_width	0	u(16)
sar_height	0	u(16)
}		
}		
overscan_info_present_flag	0	u(1)
if(overscan_info_present_flag)		
overscan_appropriate_flag	0	u(1)
video_signal_type_present_flag	0	u(1)
if(video_signal_type_present_flag) {		
video_format	0	u(3)
video_full_range_flag	0	u(1)
colour_description_present_flag	0	u(1)
if(colour_description_present_flag) {		
colour_primaries	0	u(8)
transfer_characteristics	0	u(8)
matrix_coefficients	0	u(8)
}		
}		
chroma_loc_info_present_flag	0	u(1)
if(chroma_loc_info_present_flag) {		
chroma_sample_loc_type_top_field	0	ue(v)
chroma_sample_loc_type_bottom_field	0	ue(v)
}		
timing_info_present_flag	0	u(1)
if(timing_info_present_flag) {		
num_units_in_tick	0	u(32)
time_scale	0	u(32)
fixed_frame_rate_flag	0	u(1)
}		
nal_hrd_parameters_present_flag	0	u(1)
if(nal_hrd_parameters_present_flag)		
hrd_parameters( )		
vcl_hrd_parameters_present_flag	0	u(1)
if(vcl_hrd_parameters_present_flag)		
hrd_parameters( )		
if(nal_hrd_parameters_present_flag || vcl_hrd_parameters_present_flag)		
low_delay_hrd_flag	0	u(1)
pic_struct_present_flag	0	u(1)
bitstream_restriction_flag	0	u(1)
if(bitstream_restriction_flag) {		

motion_vectors_over_pic_boundaries_flag	0	u(1)
max_bytes_per_pic_denom	0	ue(v)
max_bits_per_mb_denom	0	ue(v)
log2_max_mv_length_horizontal	0	ue(v)
log2_max_mv_length_vertical	0	ue(v)
num_reorder_frames	0	ue(v)
max_dec_frame_buffering	0	ue(v)
}		
}		

E.1.2 HRD parameters syntax

hrd_parameters( ) {	**C**	**Descriptor**
cpb_cnt_minus1	0	ue(v)
bit_rate_scale	0	u(4)
cpb_size_scale	0	u(4)
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
bit_rate_value_minus1[SchedSelIdx **]**	0	ue(v)
cpb_size_value_minus1[SchedSelIdx **]**	0	ue(v)
cbr_flag[SchedSelIdx **]**	0	u(1)
}		
initial_cpb_removal_delay_length_minus1	0	u(5)
cpb_removal_delay_length_minus1	0	u(5)
dpb_output_delay_length_minus1	0	u(5)
time_offset_length	0	u(5)
}		

E.2 VUI semantics

E.2.1 VUI parameters semantics

aspect_ratio_info_present_flag equal to 1 specifies that aspect_ratio_idc is present. aspect_ratio_info_present_flag equal to 0 specifies that aspect_ratio_idc is not present.

aspect_ratio_idc specifies the value of the sample aspect ratio of the luma samples. Table E-1 shows the meaning of the code. When aspect_ratio_idc indicates Extended_SAR, the sample aspect ratio is represented by sar_width and sar_height. When the aspect_ratio_idc syntax element is not present, aspect_ratio_idc value shall be inferred to be equal to 0.

Table E-1 – Meaning of sample aspect ratio indicator

aspect_ratio_idc	**Sample aspect ratio**	**(informative) Examples of use**
0	Unspecified	
1	1:1 ("square")	1280x720 16:9 frame without overscan 1920x1080 16:9 frame without overscan (cropped from 1920x1088) 640x480 4:3 frame without overscan
2	12:11	720x576 4:3 frame with horizontal overscan 352x288 4:3 frame without overscan
3	10:11	720x480 4:3 frame with horizontal overscan 352x240 4:3 frame without overscan
4	16:11	720x576 16:9 frame with horizontal overscan 540x576 4:3 frame with horizontal overscan
5	40:33	720x480 16:9 frame with horizontal overscan 540x480 4:3 frame with horizontal overscan
6	24:11	352x576 4:3 frame without overscan 540x576 16:9 frame with horizontal overscan
7	20:11	352x480 4:3 frame without overscan 480x480 16:9 frame with horizontal overscan
8	32:11	352x576 16:9 frame without overscan
9	80:33	352x480 16:9 frame without overscan
10	18:11	480x576 4:3 frame with horizontal overscan
11	15:11	480x480 4:3 frame with horizontal overscan
12	64:33	540x576 16:9 frame with horizontal overscan
13	160:99	540x480 16:9 frame with horizontal overscan
14..254	Reserved	
255	Extended_SAR	

sar_width indicates the horizontal size of the sample aspect ratio (in arbitrary units).

sar_height indicates the vertical size of the sample aspect ratio (in the same arbitrary units as sar_width).

sar_width and sar_height shall be relatively prime or equal to 0. When aspect_ratio_idc is equal to 0 or sar_width is equal to 0 or sar_height is equal to 0, the sample aspect ratio shall be considered unspecified by this Recommendation | International Standard.

overscan_info_present_flag equal to 1 specifies that the overscan_appropriate_flag is present. When overscan_info_present_flag is equal to 0 or is not present, the preferred display method for the video signal is unspecified.

overscan_appropriate_flag equal to 1 indicates that the cropped decoded pictures output are suitable for display using overscan. overscan_appropriate_flag equal to 0 indicates that the cropped decoded pictures output contain visually important information in the entire region out to the edges of the cropping rectangle of the picture, such that the cropped decoded pictures output should not be displayed using overscan. Instead, they should be displayed using either an exact match between the display area and the cropping rectangle, or using underscan.

NOTE – For example, overscan_appropriate_flag equal to 1 might be used for entertainment television programming, or for a live view of people in a videoconference, and overscan_appropriate_flag equal to 0 might be used for computer screen capture or security camera content.

video_signal_type_present_flag equal to 1 specifies that video_format, video_full_range_flag and colour_description_present_flag are present. video_signal_type_present_flag equal to 0, specify that video_format, video_full_range_flag and colour_description_present_flag are not present.

video_format indicates the representation of the pictures as specified in Table E-2, before being coded in accordance with this Recommendation | International Standard. When the video_format syntax element is not present, video_format value shall be inferred to be equal to 5.

Table E-2 – Meaning of video_format

video_format	Meaning
0	Component
1	PAL
2	NTSC
3	SECAM
4	MAC
5	Unspecified video format
6	Reserved
7	Reserved

video_full_range_flag indicates the black level and range of the luma and chroma signals as derived from E'_Y, E'_{PB}, and E'_{PR} analogue component signals, as follows:

If video_full_range_flag is equal to 0,

$$Y = \text{Round}(219 * E'_Y + 16) \quad \text{(E-1)}$$

$$Cb = \text{Round}(224 * E'_{PB} + 128) \quad \text{(E-2)}$$

$$Cr = \text{Round}(224 * E'_{PR} + 128) \quad \text{(E-3)}$$

Otherwise (video_full_range_flag is equal to 1),

$$Y = \text{Round}(255 * E'_Y) \quad \text{(E-4)}$$

$$Cb = \text{Round}(255 * E'_{PB} + 128) \quad \text{(E-5)}$$

$$Cr = \text{Round}(255 * E'_{PR} + 128) \quad \text{(E-6)}$$

When the video_full_range_flag syntax element is not present, video_full_range_flag value shall be inferred to be equal to 0.

colour_description_present_flag equal to 1 specifies that colour_primaries, transfer_characteristics and matrix_coefficients are present. colour_description_present_flag equal to 0 specifies that colour_primaries, transfer_characteristics and matrix_coefficients are not present.

colour_primaries indicates the chromaticity coordinates of the source primaries as specified in Table E-3 in terms of the CIE 1931 definition of x and y as specified by ISO/CIE 10527.

Table E-3 – Colour primaries

Value	Primaries
0	Reserved
1	ITU-R Recommendation BT.709 primary x y green 0.300 0.600 blue 0.150 0.060 red 0.640 0.330 white D65 0.3127 0.3290
2	Unspecified Image characteristics are unknown or as determined by the application.
3	Reserved
4	ITU-R Recommendation BT.470-2 System M primary x y green 0.21 0.71 blue 0.14 0.08 red 0.67 0.33 white C 0.310 0.316
5	ITU-R Recommendation BT.470-2 System B, G primary x y green 0.29 0.60 blue 0.15 0.06 red 0.64 0.33 white D65 0.3127 0.3290
6	Society of Motion Picture and Television Engineers 170M primary x y green 0.310 0.595 blue 0.155 0.070 red 0.630 0.340 white D65 0.3127 0.3290
7	Society of Motion Picture and Television Engineers 240M (1987) primary x y green 0.310 0.595 blue 0.155 0.070 red 0.630 0.340 white D65 0.3127 0.3290
8	Generic film (colour filters using Illuminant C) primary x y green 0.243 0.692 (Wratten 58) blue 0.145 0.049 (Wratten 47) red 0.681 0.319 (Wratten 25) white C 0.310 0.316
9-255	Reserved

When the colour_primaries syntax element is not present, the value of colour_primaries shall be inferred to be equal to 2 (the chromaticity is unspecified or is determined by the application).

transfer_characteristics indicates the opto-electronic transfer characteristic of the source picture as specified in Table E-4 as a function of a linear optical intensity input L_c with an analogue range of 0 to 1.

Table E-4 – Transfer characteristics

Value	Transfer Characteristic
0	Reserved
1	ITU-R Recommendation BT.709 $V = 1.099\ L_c^{0.45} - 0.099$ for $1 >= L_c >= 0.018$ $V = 4.500\ L_c$ for $0.018 > L_c$
2	Unspecified Image characteristics are unknown or are determined by the application.
3	Reserved
4	ITU-R Recommendation BT.470-2 System M Assumed display gamma 2.2
5	ITU-R Recommendation BT.470-2 System B, G Assumed display gamma 2.8
6	Society of Motion Picture and Television Engineers 170M $V = 1.099\ L_c^{0.45} - 0.099$ for $1 >= L_c >= 0.018$ $V = 4.500\ L_c$ for $0.018 > L_c$
7	Society of Motion Picture and Television Engineers 240M (1987) $V = 1.1115\ L_c^{0.45} - 0.1115$ for $L_c >= 0.0228$ $V = 4.0\ L_c$ for $0.0228 > L_c$
8	Linear transfer characteristics $V = L_c$
9	Logarithmic transfer characteristic (100:1 range) $V = 1.0 - \mathrm{Log10}(L_c) \div 2$ for $1 >= L_c >= 0.01$ $V = 0.0$ for $0.01 > L_c$
10	Logarithmic transfer characteristic (316.22777:1 range) $V = 1.0 - \mathrm{Log10}(L_c) \div 2.5$ for $1 >= L_c >= 0.0031622777$ $V = 0.0$ for $0.0031622777 > L_c$
11..255	Reserved

When the transfer_characteristics syntax element is not present, the value of transfer_characteristics shall be inferred to be equal to 2 (the transfer characteristics are unspecified or are determined by the application).

matrix_coefficients describes the matrix coefficients used in deriving luma and chroma signals from the green, blue, and red primaries, as specified in Table E-5.

Using the following definitions:

E'_R, E'_G, and E'_B are analogue with values in the range of 0 to 1.

White is specified as having E'_R equal to 1, E'_G equal to 1, and E'_B equal to 1.

Then:

$$E'_Y = K_R * E'_R + (1 - K_R - K_B) * E'_G + K_B * E'_B \quad \text{(E-7)}$$

$$E'_{PB} = 0.5 * (E'_B - E'_Y) \div (1 - K_B) \quad \text{(E-8)}$$

$$E'_{PR} = 0.5 * (E'_R - E'_Y) \div (1 - K_R) \quad \text{(E-9)}$$

NOTE – Then E'_Y is analogue with values in the range of 0 to 1, E'_{PB} and E'_{PR} are analogue with values in the range of -0.5 to 0.5, and white is equivalently given by $E'_Y = 1$, $E'_{PB} = 0$, $E'_{PR} = 0$.

Table E-5 – Matrix coefficients

Value	**Matrix**
0	Reserved
1	ITU-R Recommendation BT.709 $K_R = 0.2126$; $K_B = 0.0722$
2	Unspecified Image characteristics are unknown or are determined by the application.
3	Reserved
4	Federal Communications Commission $K_R = 0.30$; $K_B = 0.11$
5	ITU-R Recommendation BT.470-2 System B, G: $K_R = 0.299$; $K_B = 0.114$
6	Society of Motion Picture and Television Engineers 170M $K_R = 0.299$; $K_B = 0.114$
7	Society of Motion Picture and Television Engineers 240M (1987) $K_R = 0.212$; $K_B = 0.087$
8-255	Reserved

When the matrix_coefficients syntax element is not present, the value of matrix_coefficients shall be inferred to be equal to 2.

chroma_loc_info_present_flag equal to 1 specifies that chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are present. chroma_loc_info_present_flag equal to 0 specifies that chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are not present.

chroma_sample_loc_type_top_field and **chroma_sample_loc_type_bottom_field** specify the location of chroma samples for the top field and the bottom field as shown in Figure E-1. The value of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field shall be in the range of 0 to 5, inclusive. When the chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are not present, the values of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field shall be inferred to be equal to 0.

NOTE – When coding progressive source material, chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should have the same value.

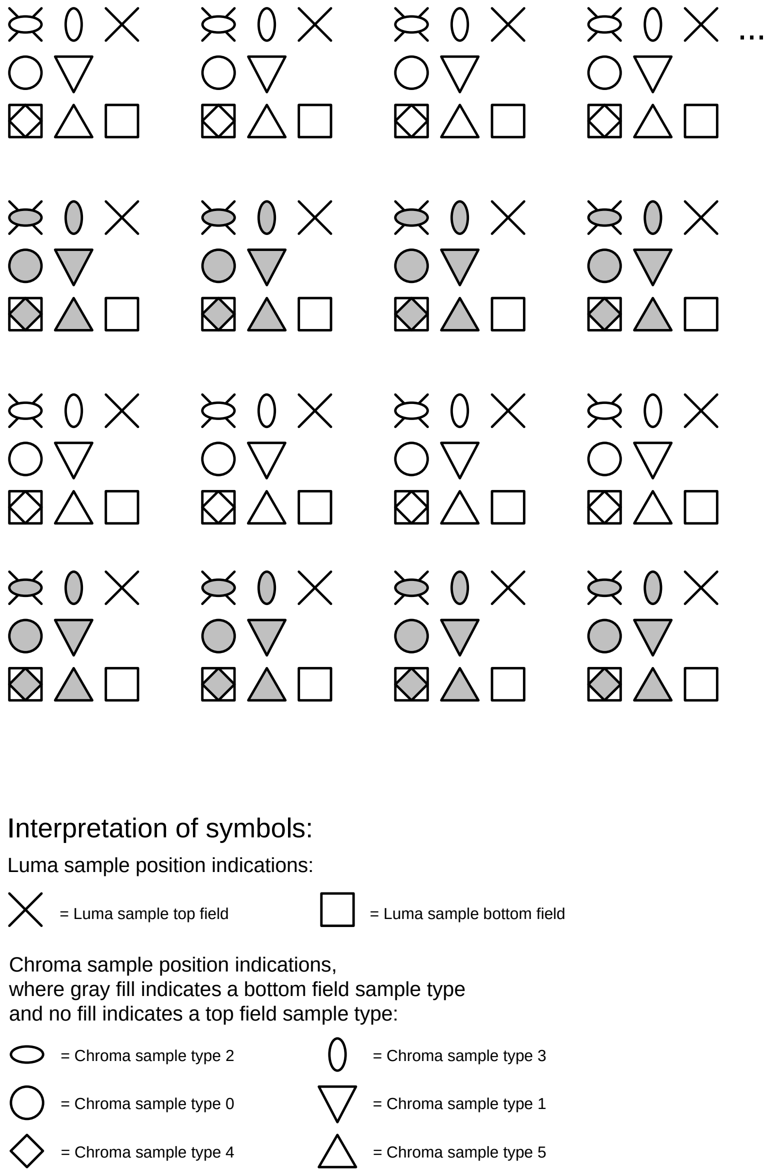


Figure E-1 – Location of chroma samples for top and bottom fields as a function of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field

timing_info_present_flag equal to 1 specifies that num_units_in_tick, time_scale and fixed_frame_rate_flag are present in the bitstream. timing_info_present_flag equal to 0 specifies that num_units_in_tick, time_scale and fixed_frame_rate_flag are not present in the bitstream.

num_units_in_tick is the number of time units of a clock operating at the frequency time_scale Hz that corresponds to one increment (called a clock tick) of a clock tick counter. num_units_in_tick shall be greater than 0. A clock tick is the minimum interval of time that can be represented in the coded data. For example, when the clock frequency of a video signal is 30000 ÷ 1001 Hz, time_scale may be 30 000 and num_units_in_tick may be 1001. See Equation C-1.

time_scale is the number of time units that pass in one second. For example, a time coordinate system that measures time using a 27 MHz clock has a time_scale of 27 000 000. time_scale shall be greater than 0.

fixed_frame_rate_flag equal to 1 indicates that the temporal distance between the HRD output times of any two consecutive pictures in output order is constrained as follows. fixed_frame_rate_flag equal to 0 indicates that no such constraints apply to the temporal distance between the HRD output times of any two consecutive pictures in output order.

When fixed_frame_rate_flag is equal to 1, for all n where n indicates the n-th picture in output order and picture n is not the last picture in the bitstream in output order, the value of $\Delta t_{fi,dpb}(n)$ as specified in Equation C-13, shall obey the following constraint,

$$\Delta t_{fi,dpb}(n) == \Delta t_{o,dpb}(n) \div \text{DeltaTfiDivisor} \qquad \text{(E-10)}$$

where DeltaTfiDivisor is specified by Table E-6 based on the value of pic_struct_present_flag, field_pic_flag, and pic_struct for picture n. Entries marked "-" in Table E-6 indicate a lack of dependence of DeltaTfiDivisor on the corresponding syntax element.

The value computed for $\Delta t_{fi,dpb}(n)$ shall be the same for all n > 0. and equal to num_units_in_tick ÷ time_scale.

Table E-6 – Divisor for computation of $\Delta t_{fi,dpb}(n)$

pic_struct_present_flag	field_pic_flag	pic_struct	DeltaTfiDivisor
0	1	-	1
1	-	1	1
1	-	2	1
0	0	-	2
1	-	0	2
1	-	3	2
1	-	4	2
1	-	5	3
1	-	6	3
1	-	7	4
1	-	8	6

nal_hrd_parameters_present_flag equal to 1 specifies that NAL HRD parameters (pertaining to Type II bitstream conformance) are present. nal_hrd_parameters_present_flag equal to 0 specifies that NAL HRD parameters are not present.

NOTE – When nal_hrd_parameters_present_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the NAL HRD parameters, including the NAL sequence HRD parameter information and all buffering period and picture timing SEI messages, by some means not specified in this Recommendation | International Standard

When nal_hrd_parameters_present_flag is equal to 1, NAL HRD parameters (subclauses E.1.2 and E.2.2) immediately follow the flag.

If any of the following is true, the value of NalHrdBpPresentFlag shall be set equal to 1.

– nal_hrd_parameters_present_flag is present in the bitstream and is equal to 1
– the need for presence of buffering periods for NAL HRD operation to be present in the bitstream in buffering period SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.

Otherwise, the value of NalHrdBpPresentFlag shall be set equal to 0.

vcl_hrd_parameters_present_flag equal to 1 specifies that VCL HRD parameters (pertaining to all bitstream conformance) are present. vcl_hrd_parameters_present_flag equal to 0 specifies that VCL HRD parameters are not present.

NOTE – When vcl_hrd_parameters_present_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the VCL HRD parameters and all buffering period and picture timing SEI messages, by some means not specified in this Recommendation | International Standard

When vcl_hrd_parameters_present_flag is equal to 1, VCL HRD parameters (subclauses E.1.2 and E.2.2) immediately follow the flag.

If any of the following is true, the value of VclHrdBpPresentFlag shall be set equal to 1.

– nal_hrd_parameters_present_flag is present in the bitstream and is equal to 1
– the need for presence of buffering periods for VCL HRD operation to be present in the bitstream in buffering period SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.

Otherwise, the value of VclHrdBpPresentFlag shall be set equal to 0.

If any of the following is true, the value of CpbDpbDelaysPresentFlag shall be set equal to 1.

– nal_hrd_parameters_present_flag is present in the bitstream and is equal to 1
– vcl_hrd_parameters_present_flag is present in the bitstream and is equal to 1

– the need for presence of CPB and DPB output delays to be present in the bitstream in picture timing SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.

Otherwise, the value of CpbDpbDelaysPresentFlag shall be set equal to 0.

low_delay_hrd_flag specifies the HRD operational mode as specified in Annex C. When fixed_frame_rate_flag is equal to 1, low_delay_hrd_flag shall be equal to 0.

NOTE - When low_delay_hrd_flag is equal to 1, "big pictures" that violate the nominal CPB removal times due to the number of bits used by an access unit are permitted. It is expected, but not required, that such "big pictures" occur only occasionally.

pic_struct_present_flag equal to 1 specifies picture timing SEI messages (subclause D.2.2) are present that include the pic_struct syntax element. pic_struct_present_flag equal to 0 specifies that the pic_struct syntax element is not present in picture timing SEI messages.

bitstream_restriction_flag equal to 1, specifies that the following sequence bitstream restriction parameters are present. bitstream_restriction_flag equal to 0, specifies that the following sequence bitstream restriction parameters are not present.

motion_vectors_over_pic_boundaries_flag equal to 0 indicates that no sample outside the picture boundaries and no sample at a fractional sample position whose value is derived using one or more samples outside the picture boundaries is used to inter predict any sample. motion_vectors_over_pic_boundaries_flag equal to 1 indicates that one or more samples outside picture boundaries may be used in inter prediction. When the motion_vectors_over_pic_boundaries_flag syntax element is not present, motion_vectors_over_pic_boundaries_flag value shall be inferred to be equal to 1.

max_bytes_per_pic_denom indicates a number of bytes not exceeded by the sum of the sizes of the VCL NAL units associated with any coded picture in the sequence.

The number of bytes that represent a picture in the NAL unit stream is specified for this purpose as the total number of bytes of VCL NAL unit data (i.e., the total of the NumBytesInNALunit variables for the VCL NAL units) for the picture. The value of max_bytes_per_pic_denom shall be in the range of 0 to 16, inclusive.

- If max_bytes_per_pic_denom is equal to 0, no limits are indicated.
- Otherwise (max_bytes_per_pic_denom is not equal to 0), no coded picture shall be represented in the sequence by more than the following number of bytes.

$$(\text{PicSizeInMbs} * 256 * \text{ChromaFormatFactor}) \div \text{max\_bytes\_per\_pic\_denom} \qquad \text{(E-11)}$$

When the max_bytes_per_pic_denom syntax element is not present, the value of max_bytes_per_pic_denom shall be inferred to be equal to 2.

max_bits_per_mb_denom indicates the maximum number of coded bits of macroblock_layer( ) data for any macroblock in any picture of the sequence. The value of max_bits_per_mb_denom shall be in the range of 0 to 16, inclusive.

- If max_bits_per_mb_denom is equal to 0, no limit is specified.
- Otherwise (max_bits_per_mb_denom is not equal to 0), no coded macroblock_layer( ) shall be represented in the bitstream by more than the following number of bits.

$$(2048 * \text{ChromaFormatFactor} + 128) \div \text{max\_bits\_per\_mb\_denom} \qquad \text{(E-12)}$$

Depending on entropy_coding_mode_flag, the bits of macroblock_layer( ) data are counted as follows.

- If entropy_coding_mode_flag is equal to 0, the number of bits of macroblock_layer( ) data is given by the number of bits in the macroblock_layer( ) syntax structure for a macroblock.
- Otherwise (entropy_coding_mode_flag is equal to 1), the number of bits of macroblock_layer( ) data for a macroblock is given by the number of times read_bits(1) is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing the macroblock_layer( ) associated with the macroblock.

When the max_bits_per_mb_denom is not present, the value of max_bits_per_mb_denom shall be inferred to be equal to 1.

log2_max_mv_length_horizontal and **log2_max_mv_length_vertical** indicate the maximum absolute value of a decoded horizontal and vertical motion vector component, respectively, in ¼ luma sample units, for all pictures in the sequence. A value of n asserts that no absolute value of a motion vector component is larger than 2^n units of ¼ luma sample displacement. The value of log2_max_mv_length_horizontal shall be in the range of 0 to 16, inclusive. The value of log2_max_mv_length_vertical shall be in the range of 0 to 16, inclusive. When log2_max_mv_length_horizontal is not present, the values of log2_max_mv_length_horizontal and log2_max_mv_length_vertical shall be inferred to be equal to 16.

NOTE - The maximum absolute value of a decoded vertical or horizontal motion vector component is also constrained by profile and level limits as specified in Annex A.

num_reorder_frames indicates the maximum amount of frames, complementary field pairs, or non-paired fields that precede any frame, complementary field pair, or non-paired fields in the sequence in decoding order and follow it in output order. The value of num_reorder_frames shall be in the range of 0 to max_dec_frame_buffering, inclusive. When the num_reorder_frames syntax element is not present, the value of num_reorder_frames value shall be inferred to be equal to max_dec_frame_buffering.

max_dec_frame_buffering specifies the required size of the HRD decoded picture buffer (DPB) in units of frame buffers. The sequence shall not require a decoded picture buffer with size of more than max_dec_frame_buffering frame buffers to enable the output of decoded pictures at the output times specified by dpb_output_delay of the picture timing SEI messages. The value of max_dec_frame_buffering shall be in the range of 0 to MaxDpbSize, inclusive (as specified in subclause A.3.1). When the max_dec_frame_buffering syntax element is not present, the value of max_dec_frame_buffering shall be inferred to be equal to MaxDpbSize.

E.2.2 HRD parameters semantics

cpb_cnt_minus1 plus 1 specifies the number of alternative CPB specifications in the bitstream. The value of cpb_cnt_minus1 shall be in the range of 0 to 31, inclusive. When low_delay_hrd_flag is equal to 1, cpb_cnt_minus1 shall be equal to 0. When cpb_cnt_minus1 is not present, it shall be inferred to be equal to 0.

bit_rate_scale (together with bit_rate_value_minus1[SchedSelIdx]) specifies the maximum input bit rate of the SchedSelIdx-th CPB.

cpb_size_scale (together with cpb_size_value_minus1[SchedSelIdx]) specifies the CPB size of the SchedSelIdx-th CPB.

bit_rate_value_minus1[SchedSelIdx **]** (together with bit_rate_scale) specifies the maximum input bit rate for the SchedSelIdx-th CPB. bit_rate_value_minus1[SchedSelIdx] shall be in the range of 0 to $2^{32} - 2$, inclusive. For any SchedSelIdx > 0, bit_rate_value_minus1[SchedSelIdx] shall be greater than bit_rate_value_minus1[SchedSelIdx - 1]. The bit rate in bits per second is given by

$$\text{BitRate[SchedSelIdx]} = (\text{ bit\_rate\_value\_minus1[SchedSelIdx]} + 1\text{ }) * 2^{(6 + \text{bit\_rate\_scale})} \quad \text{(E-13)}$$

When the bit_rate_value_minus1[SchedSelIdx] syntax element is not present, BitRate[SchedSelIdx] shall be inferred to be equal to 1000 * MaxBR for VCL HRD parameters.

When the bit_rate_value_minus1[SchedSelIdx] syntax element is not present, BitRate[SchedSelIdx] shall be inferred to be equal to 1200 * MaxBR for NAL HRD parameters.

cpb_size_value_minus1[SchedSelIdx **]** is used together with cpb_size_scale to define the SchedSelIdx-th CPB size. cpb_size_value_minus1[SchedSelIdx] shall be in the range of 0 to $2^{32} - 2$, inclusive. For any SchedSelIdx greater than 0, cpb_size_value_minus1[SchedSelIdx] shall be less than or equal to cpb_size_value_minus1[SchedSelIdx -1].

The CPB size in bits is given by

$$\text{CpbSize[SchedSelIdx]} = (\text{ cpb\_size\_value\_minus1[SchedSelIdx]} + 1\text{ }) * 2^{(4 + \text{cpb\_size\_scale})} \quad \text{(E-14)}$$

When the cpb_size_value_minus1[SchedSelIdx] syntax element is not present, CpbSize[SchedSelIdx] shall be inferred to be equal to 1000 * MaxCPB for VCL HRD parameters.

When the cpb_size_value_minus1[SchedSelIdx] syntax element is not present, CpbSize[SchedSelIdx] shall be inferred to be equal to 1200 * MaxCPB for NAL HRD parameters.

For VCL HRD parameters, there shall be at least one value of SchedSelIdx for which BitRate[SchedSelIdx] <= 1000*MaxBR and CpbSize[SchedSelIdx] <= 1000 * MaxCPB (as specified in subclause A.3.1).

For NAL HRD parameters, there shall be at least one value of SchedSelIdx for which CpbSize[SchedSelIdx] <= 1200*MaxCPB and BitRate[SchedSelIdx] <= 1200*MaxBR.

cbr_flag[SchedSelIdx **]** equal to 0 specifies that to decode this bitstream by the HRD using the SchedSelIdx-th CPB specification, the hypothetical stream delivery scheduler (HSS) operates in an intermittent bit rate mode. cbr_flag[SchedSelIdx] equal to 1 specifies that the HSS operates in a constant bit rate (CBR) mode. When the cbr_flag[SchedSelIdx] syntax element is not present, the value of cbr_flag shall be inferred to be equal to 0.

initial_cpb_removal_delay_length_minus1 specifies the length in bits of the initial_cpb_removal_delay and initial_cpb_removal_delay_offset syntax elements of the buffering period SEI message. The length of initial_cpb_removal_delay and of initial_cpb_removal_delay_offset is initial_cpb_removal_delay_length_minus1 + 1.

cpb_removal_delay_length_minus1 specifies the length in bits of the cpb_removal_delay syntax element. The length of the cpb_removal_delay syntax element of the picture timing SEI message is cpb_removal_delay_length_minus1 + 1.

dpb_output_delay_length_minus1 specifies the length in bits of the dpb_output_delay syntax element. The length of the dpb_output_delay syntax element of the picture timing SEI message is dpb_output_delay_length_minus1 + 1.

time_offset_length greater than 0 specifies the length in bits of the time_offset syntax element. time_offset_length equal to 0 specifies that the time_offset syntax element is not present.