AS ISO/IEC 14496.10—2006
ISO/IEC 14496-10:2005

AS ISO/IEC 14496.10—2006

Australian Standard™

Information technology—Coding of audio-visual objects

Part 10: Advanced Video Coding

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

This Australian Standard was prepared by Committee IT-029, Coded Representation of Picture, Audio and Multimedia/Hypermedia Information. It was approved on behalf of the Council of Standards Australia on 12 May 2006.
This Standard was published on 2 June 2006.

The following are represented on Committee IT-029:

Australian Broadcasting Authority (ABA)
Australian Broadcasting Corporation (ABC)
Australian Consumers Association
Australian Subscription Television
CSIRO Mathematical & Information Services
Department of Defence (Australia)
Free TV Australia
School of Computer Science and Mathematics
Victoria University of Technology
Special Broadcasting Service (SBS)
The University of New South Wales
University of Wollongong

Keeping Standards up-to-date

Standards are living documents which reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued. Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments which may have been published since the Standard was purchased.

Detailed information about Standards can be found by visiting the Standards Web Shop at www.standards.com.au and looking up the relevant Standard in the on-line catalogue.

Alternatively, the printed Catalogue provides information current at 1 January each year, and the monthly magazine, *The Global Standard*, has a full listing of revisions and amendments published each month.

Australian Standards™ and other products and services developed by Standards Australia are published and distributed under contract by SAI Global, which operates the Standards Web Shop.

We also welcome suggestions for improvement in our Standards, and especially encourage readers to notify us immediately of any apparent inaccuracies or ambiguities. Contact us via email at mail@standards.org.au, or write to the Chief Executive, Standards Australia, GPO Box 476, Sydney, NSW 2001.

This Standard was issued in draft form for comment as DR 06118.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Australian Standard™

Information technology—Coding of audio-visual objects

Part 10: Advanced Video Coding

First published as AS ISO/IEC 14496.10—2006.

COPYRIGHT

© Standards Australia

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher.

Published by Standards Australia GPO Box 476, Sydney, NSW 2001, Australia

ISBN 0 7337 7471 7

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

PREFACE

This Standard was prepared by the Standards Australia Committee IT-029, Coded Representation of Picture, Audio and Multimedia/Hypermedia Information.

This Standard is identical with, and has been reproduced from ISO/IEC 14496-10:2005, *Information technology—Coding of audio-visual objects,* Part 10: *Advanced Video Coding*.

The objective of this Standard is to provide the multimedia community with the mechanism of 'Advanced Video Coding', which facilitates video data interchange among different applications, including services using video over DSL networks and handheld devices with the ultimate goal of providing bandwidth cost savings for the community.

Standards Australia is in the process of adopting most of ISO/IEC 14496 series Standards. Refer to the website for current list.

As this Standard is reproduced from an international standard, the following applies:

(a) Its number appears on the cover and title page while the international standard number appears only on the cover

(b) In the source text 'this part of ISO/IEC 14499' should read 'this Australian Standard'.

(c) A full point substitutes for a comma when referring to a decimal marker.

None of the normative references in the source document have been adopted as Australian or Australian/New Zealand Standards.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

CONTENTS

Page

0 Introduction xiv
0.1 Prologue xiv
0.2 Purpose xiv
0.3 Applications xiv
0.4 Publication and versions of this International Standard xiv
0.5 Profiles and levels xv
0.6 Overview of the design characteristics xv
0.6.1 Predictive coding xv
0.6.2 Coding of progressive and interlaced video xvi
0.6.3 Picture partitioning into macroblocks and smaller partitions xvi
0.6.4 Spatial redundancy reduction xvi
0.7 How to read this specification xvi
1 Scope 1
2 Normative references 1
3 Definitions 1
4 Abbreviations 9
5 Conventions 10
5.1 Arithmetic operators 10
5.2 Logical operators 10
5.3 Relational operators 11
5.4 Bit-wise operators 11
5.5 Assignment operators 11
5.6 Range notation 11
5.7 Mathematical functions 11
5.8 Variables, syntax elements, and tables 12
5.9 Text description of logical operations 13
5.10 Processes 14
6 Source, coded, decoded and output data formats, scanning processes, and neighbouring relationships 14
6.1 Bitstream formats 14
6.2 Source, decoded, and output picture formats 15
6.3 Spatial subdivision of pictures and slices 20
6.4 Inverse scanning processes and derivation processes for neighbours 20
6.4.1 Inverse macroblock scanning process 21
6.4.2 Inverse macroblock partition and sub-macroblock partition scanning process 21
6.4.2.1 Inverse macroblock partition scanning process 22
6.4.2.2 Inverse sub-macroblock partition scanning process 22
6.4.3 Inverse 4x4 luma block scanning process 23
6.4.4 Inverse 8x8 luma block scanning process 23
6.4.5 Derivation process of the availability for macroblock addresses 23
6.4.6 Derivation process for neighbouring macroblock addresses and their availability 24
6.4.7 Derivation process for neighbouring macroblock addresses and their availability in MBAFF frames 24
6.4.8 Derivation processes for neighbouring macroblocks, blocks, and partitions 25
6.4.8.1 Derivation process for neighbouring macroblocks 26
6.4.8.2 Derivation process for neighbouring 8x8 luma block 26
6.4.8.3 Derivation process for neighbouring 4x4 luma blocks 27
6.4.8.4 Derivation process for neighbouring 4x4 chroma blocks 28
6.4.8.5 Derivation process for neighbouring partitions 28
6.4.9 Derivation process for neighbouring locations 30
6.4.9.1 Specification for neighbouring locations in fields and non-MBAFF frames 30
6.4.9.2 Specification for neighbouring locations in MBAFF frames 31

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

7 Syntax and semantics 33
7.1 Method of describing syntax in tabular form 33
7.2 Specification of syntax functions, categories, and descriptors 34
7.3 Syntax in tabular form 36
7.3.1 NAL unit syntax 36
7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax 37
7.3.2.1 Sequence parameter set RBSP syntax 37
7.3.2.1.1 Scaling list syntax 38
7.3.2.1.2 Sequence parameter set extension RBSP syntax 39
7.3.2.2 Picture parameter set RBSP syntax 39
7.3.2.3 Supplemental enhancement information RBSP syntax 40
7.3.2.3.1 Supplemental enhancement information message syntax 41
7.3.2.4 Access unit delimiter RBSP syntax 41
7.3.2.5 End of sequence RBSP syntax 41
7.3.2.6 End of stream RBSP syntax 41
7.3.2.7 Filler data RBSP syntax 42
7.3.2.8 Slice layer without partitioning RBSP syntax 42
7.3.2.9 Slice data partition RBSP syntax 42
7.3.2.9.1 Slice data partition A RBSP syntax 42
7.3.2.9.2 Slice data partition B RBSP syntax 42
7.3.2.9.3 Slice data partition C RBSP syntax 43
7.3.2.10 RBSP slice trailing bits syntax 43
7.3.2.11 RBSP trailing bits syntax 43
7.3.3 Slice header syntax 44
7.3.3.1 Reference picture list reordering syntax 45
7.3.3.2 Prediction weight table syntax 46
7.3.3.3 Decoded reference picture marking syntax 47
7.3.4 Slice data syntax 48
7.3.5 Macroblock layer syntax 49
7.3.5.1 Macroblock prediction syntax 50
7.3.5.2 Sub-macroblock prediction syntax 51
7.3.5.3 Residual data syntax 52
7.3.5.3.1 Residual block CAVLC syntax 53
7.3.5.3.2 Residual block CABAC syntax 55
7.4 Semantics 56
7.4.1 NAL unit semantics 56
7.4.1.1 Encapsulation of an SODB within an RBSP (informative) 58
7.4.1.2 Order of NAL units and association to coded pictures, access units, and video sequences 59
7.4.1.2.1 Order of sequence and picture parameter set RBSPs and their activation 59
7.4.1.2.2 Order of access units and association to coded video sequences 60
7.4.1.2.3 Order of NAL units and coded pictures and association to access units 61
7.4.1.2.4 Detection of the first VCL NAL unit of a primary coded picture 62
7.4.1.2.5 Order of VCL NAL units and association to coded pictures 63
7.4.2 Raw byte sequence payloads and RBSP trailing bits semantics 63
7.4.2.1 Sequence parameter set RBSP semantics 63
7.4.2.1.1 Scaling list semantics 68
7.4.2.1.2 Sequence parameter set extension RBSP semantics 69
7.4.2.2 Picture parameter set RBSP semantics 70
7.4.2.3 Supplemental enhancement information RBSP semantics 73
7.4.2.3.1 Supplemental enhancement information message semantics 73
7.4.2.4 Access unit delimiter RBSP semantics 73
7.4.2.5 End of sequence RBSP semantics 73
7.4.2.6 End of stream RBSP semantics 74
7.4.2.7 Filler data RBSP semantics 74
7.4.2.8 Slice layer without partitioning RBSP semantics 74
7.4.2.9 Slice data partition RBSP semantics 74
7.4.2.9.1 Slice data partition A RBSP semantics 74
7.4.2.9.2 Slice data partition B RBSP semantics 74
7.4.2.9.3 Slice data partition C RBSP semantics 74

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

7.4.2.10 RBSP slice trailing bits semantics ... 75
7.4.2.11 RBSP trailing bits semantics ... 75
7.4.3 Slice header semantics ... 75
7.4.3.1 Reference picture list reordering semantics ... 81
7.4.3.2 Prediction weight table semantics ... 82
7.4.3.3 Decoded reference picture marking semantics ... 82
7.4.4 Slice data semantics ... 85
7.4.5 Macroblock layer semantics ... 86
7.4.5.1 Macroblock prediction semantics ... 93
7.4.5.2 Sub-macroblock prediction semantics ... 94
7.4.5.3 Residual data semantics ... 96
7.4.5.3.1 Residual block CAVLC semantics ... 97
7.4.5.3.2 Residual block CABAC semantics ... 97
8 Decoding process ... 98
8.1 NAL unit decoding process ... 99
8.2 Slice decoding process ... 99
8.2.1 Decoding process for picture order count ... 99
8.2.1.1 Decoding process for picture order count type 0 ... 101
8.2.1.2 Decoding process for picture order count type 1 ... 102
8.2.1.3 Decoding process for picture order count type 2 ... 103
8.2.2 Decoding process for macroblock to slice group map ... 104
8.2.2.1 Specification for interleaved slice group map type ... 105
8.2.2.2 Specification for dispersed slice group map type ... 105
8.2.2.3 Specification for foreground with left-over slice group map type ... 105
8.2.2.4 Specification for box-out slice group map types ... 106
8.2.2.5 Specification for raster scan slice group map types ... 106
8.2.2.6 Specification for wipe slice group map types ... 106
8.2.2.7 Specification for explicit slice group map type ... 107
8.2.2.8 Specification for conversion of map unit to slice group map to macroblock to slice group map ... 107
8.2.3 Decoding process for slice data partitioning ... 107
8.2.4 Decoding process for reference picture lists construction ... 108
8.2.4.1 Decoding process for picture numbers ... 108
8.2.4.2 Initialisation process for reference picture lists ... 109
8.2.4.2.1 Initialisation process for the reference picture list for P and SP slices in frames ... 109
8.2.4.2.2 Initialisation process for the reference picture list for P and SP slices in fields ... 110
8.2.4.2.3 Initialisation process for reference picture lists for B slices in frames ... 110
8.2.4.2.4 Initialisation process for reference picture lists for B slices in fields ... 111
8.2.4.2.5 Initialisation process for reference picture lists in fields ... 112
8.2.4.3 Reordering process for reference picture lists ... 113
8.2.4.3.1 Reordering process of reference picture lists for short-term reference pictures ... 113
8.2.4.3.2 Reordering process of reference picture lists for long-term reference pictures ... 114
8.2.5 Decoded reference picture marking process ... 114
8.2.5.1 Sequence of operations for decoded reference picture marking process ... 115
8.2.5.2 Decoding process for gaps in frame_num ... 115
8.2.5.3 Sliding window decoded reference picture marking process ... 116
8.2.5.4 Adaptive memory control decoded reference picture marking process ... 116
8.2.5.4.1 Marking process of a short-term reference picture as “unused for reference” ... 116
8.2.5.4.2 Marking process of a long-term reference picture as “unused for reference” ... 117
8.2.5.4.3 Assignment process of a LongTermFrameIdx to a short-term reference picture ... 117
8.2.5.4.4 Decoding process for MaxLongTermFrameIdx ... 117
8.2.5.4.5 Marking process of all reference pictures as “unused for reference” and setting MaxLongTermFrameIdx to “no long-term frame indices” ... 118
8.2.5.4.6 Process for assigning a long-term frame index to the current picture ... 118
8.3 Intra prediction process ... 118
8.3.1 Intra_4x4 prediction process for luma samples ... 119
8.3.1.1 Derivation process for the Intra4x4PredMode ... 119
8.3.1.2 Intra_4x4 sample prediction ... 121
8.3.1.2.1 Specification of Intra_4x4_Vertical prediction mode ... 122
8.3.1.2.2 Specification of Intra_4x4_Horizontal prediction mode ... 122
8.3.1.2.3 Specification of Intra_4x4_DC prediction mode ... 122

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

8.3.1.2.4 Specification of Intra_4x4_Diagonal_Down_Left prediction mode 123
8.3.1.2.5 Specification of Intra_4x4_Diagonal_Down_Right prediction mode 123
8.3.1.2.6 Specification of Intra_4x4_Vertical_Right prediction mode 123
8.3.1.2.7 Specification of Intra_4x4_Horizontal_Down prediction mode 124
8.3.1.2.8 Specification of Intra_4x4_Vertical_Left prediction mode 124
8.3.1.2.9 Specification of Intra_4x4_Horizontal_Up prediction mode 124
8.3.2 Intra_8x8 prediction process for luma samples 125
8.3.2.1 Derivation process for Intra8x8PredMode 125
8.3.2.2 Intra_8x8 sample prediction 127
8.3.2.2.1 Reference sample filtering process for Intra_8x8 sample prediction 128
8.3.2.2.2 Specification of Intra_8x8_Vertical prediction mode 129
8.3.2.2.3 Specification of Intra_8x8_Horizontal prediction mode 129
8.3.2.2.4 Specification of Intra_8x8_DC prediction mode 129
8.3.2.2.5 Specification of Intra_8x8_Diagonal_Down_Left prediction mode 130
8.3.2.2.6 Specification of Intra_8x8_Diagonal_Down_Right prediction mode 130
8.3.2.2.7 Specification of Intra_8x8_Vertical_Right prediction mode 130
8.3.2.2.8 Specification of Intra_8x8_Horizontal_Down prediction mode 131
8.3.2.2.9 Specification of Intra_8x8_Vertical_Left prediction mode 131
8.3.2.2.10 Specification of Intra_8x8_Horizontal_Up prediction mode 131
8.3.3 Intra_16x16 prediction process for luma samples 132
8.3.3.1 Specification of Intra_16x16_Vertical prediction mode 133
8.3.3.2 Specification of Intra_16x16_Horizontal prediction mode 133
8.3.3.3 Specification of Intra_16x16_DC prediction mode 133
8.3.3.4 Specification of Intra_16x16_Plane prediction mode 133
8.3.4 Intra prediction process for chroma samples 134
8.3.4.1 Specification of Intra_Chroma_DC prediction mode 134
8.3.4.2 Specification of Intra_Chroma_Horizontal prediction mode 136
8.3.4.3 Specification of Intra_Chroma_Vertical prediction mode 136
8.3.4.4 Specification of Intra_Chroma_Plane prediction mode 136
8.3.5 Sample construction process for I_PCM macroblocks 137
8.4 Inter prediction process 137
8.4.1 Derivation process for motion vector components and reference indices 139
8.4.1.1 Derivation process for luma motion vectors for skipped macroblocks in P and SP slices 140
8.4.1.2 Derivation process for luma motion vectors for B_Skip, B_Direct_16x16, and B_Direct_8x8 141
8.4.1.2.1 Derivation process for the co-located 4x4 sub-macroblock partitions 141
8.4.1.2.2 Derivation process for spatial direct luma motion vector and reference index prediction mode 144
8.4.1.2.3 Derivation process for temporal direct luma motion vector and reference index prediction mode 146
8.4.1.3 Derivation process for luma motion vector prediction 148
8.4.1.3.1 Derivation process for median luma motion vector prediction 149
8.4.1.3.2 Derivation process for motion data of neighbouring partitions 150
8.4.1.4 Derivation process for chroma motion vectors 151
8.4.2 Decoding process for Inter prediction samples 151
8.4.2.1 Reference picture selection process 152
8.4.2.2 Fractional sample interpolation process 153
8.4.2.2.1 Luma sample interpolation process 154
8.4.2.2.2 Chroma sample interpolation process 157
8.4.2.3 Weighted sample prediction process 158
8.4.2.3.1 Default weighted sample prediction process 158
8.4.2.3.2 Weighted sample prediction process 159
8.5 Transform coefficient decoding process and picture construction process prior to deblocking filter process .. 161
8.5.1 Specification of transform decoding process for 4x4 luma residual blocks 162
8.5.2 Specification of transform decoding process for luma samples of Intra_16x16 macroblock prediction mode 162
8.5.3 Specification of transform decoding process for 8x8 luma residual blocks 163
8.5.4 Specification of transform decoding process for chroma samples 164
8.5.5 Inverse scanning process for transform coefficients 166
8.5.6 Inverse scanning process for 8x8 luma transform coefficients 167
8.5.7 Derivation process for the chroma quantisation parameters and scaling function 169

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

8.5.8 Scaling and transformation process for luma DC transform coefficients for Intra_16x16 macroblock type ... 171
8.5.9 Scaling and transformation process for chroma DC transform coefficients ... 171
8.5.10 Scaling and transformation process for residual 4x4 blocks ... 173
8.5.11 Scaling and transformation process for residual 8x8 luma blocks ... 175
8.5.12 Picture construction process prior to deblocking filter process ... 178
8.5.13 Residual colour transform process ... 179
8.6 Decoding process for P macroblocks in SP slices or SI macroblocks ... 179
8.6.1 SP decoding process for non-switching pictures ... 180
8.6.1.1 Luma transform coefficient decoding process ... 180
8.6.1.2 Chroma transform coefficient decoding process ... 181
8.6.2 SP and SI slice decoding process for switching pictures ... 182
8.6.2.1 Luma transform coefficient decoding process ... 183
8.6.2.2 Chroma transform coefficient decoding process ... 183
8.7 Deblocking filter process ... 184
8.7.1 Filtering process for block edges ... 188
8.7.2 Filtering process for a set of samples across a horizontal or vertical block edge ... 189
8.7.2.1 Derivation process for the luma content dependent boundary filtering strength ... 190
8.7.2.2 Derivation process for the thresholds for each block edge ... 191
8.7.2.3 Filtering process for edges with bS less than 4 ... 193
8.7.2.4 Filtering process for edges for bS equal to 4 ... 194
9 Parsing process ... 195
9.1 Parsing process for Exp-Golomb codes ... 195
9.1.1 Mapping process for signed Exp-Golomb codes ... 197
9.1.2 Mapping process for coded block pattern ... 197
9.2 CAVLC parsing process for transform coefficient levels ... 200
9.2.1 Parsing process for total number of transform coefficient levels and trailing ones ... 201
9.2.2 Parsing process for level information ... 205
9.2.2.1 Parsing process for level_prefix ... 206
9.2.3 Parsing process for run information ... 206
9.2.4 Combining level and run information ... 209
9.3 CABAC parsing process for slice data ... 209
9.3.1 Initialisation process ... 210
9.3.1.1 Initialisation process for context variables ... 211
9.3.1.2 Initialisation process for the arithmetic decoding engine ... 222
9.3.2 Binarization process ... 222
9.3.2.1 Unary (U) binarization process ... 224
9.3.2.2 Truncated unary (TU) binarization process ... 224
9.3.2.3 Concatenated unary/ k-th order Exp-Golomb (UEGk) binarization process ... 225
9.3.2.4 Fixed-length (FL) binarization process ... 225
9.3.2.5 Binarization process for macroblock type and sub-macroblock type ... 226
9.3.2.6 Binarization process for coded block pattern ... 229
9.3.2.7 Binarization process for mb_qp_delta ... 229
9.3.3 Decoding process flow ... 230
9.3.3.1 Derivation process for ctxIdx ... 230
9.3.3.1.1 Assignment process of ctxIdxInc using neighbouring syntax elements ... 232
9.3.3.1.1.1 Derivation process of ctxIdxInc for the syntax element mb_skip_flag ... 232
9.3.3.1.1.2 Derivation process of ctxIdxInc for the syntax element mb_field_decoding_flag ... 232
9.3.3.1.1.3 Derivation process of ctxIdxInc for the syntax element mb_type ... 233
9.3.3.1.1.4 Derivation process of ctxIdxInc for the syntax element coded_block_pattern ... 233
9.3.3.1.1.5 Derivation process of ctxIdxInc for the syntax element mb_qp_delta ... 234
9.3.3.1.1.6 Derivation process of ctxIdxInc for the syntax elements ref_idx_l0 and ref_idx_l1 ... 234
9.3.3.1.1.7 Derivation process of ctxIdxInc for the syntax elements mvd_l0 and mvd_l1 ... 235
9.3.3.1.1.8 Derivation process of ctxIdxInc for the syntax element intra_chroma_pred_mode ... 237
9.3.3.1.1.9 Derivation process of ctxIdxInc for the syntax element coded_block_flag ... 237
9.3.3.1.1.10 Derivation process of ctxIdxInc for the syntax element transform_size_8x8_flag ... 238
9.3.3.1.2 Assignment process of ctxIdxInc using prior decoded bin values ... 239
9.3.3.1.3 Assignment process of ctxIdxInc for syntax elements significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1 ... 239
9.3.3.2 Arithmetic decoding process ... 242

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

9.3.3.2.1 Arithmetic decoding process for a binary decision 243
9.3.3.2.1.1 State transition process 243
9.3.3.2.2 Renormalization process in the arithmetic decoding engine 246
9.3.3.2.3 Bypass decoding process for binary decisions 247
9.3.3.2.4 Decoding process for binary decisions before termination 247
9.3.4 Arithmetic encoding process (informative) 248
9.3.4.1 Initialisation process for the arithmetic encoding engine (informative) 248
9.3.4.2 Encoding process for a binary decision (informative) 248
9.3.4.3 Renormalization process in the arithmetic encoding engine (informative) 249
9.3.4.4 Bypass encoding process for binary decisions (informative) 251
9.3.4.5 Encoding process for a binary decision before termination (informative) 252
9.3.4.6 Byte stuffing process (informative) 254

Annex A Profiles and levels 255
A.1 Requirements on video decoder capability 255
A.2 Profiles 255
A.2.1 Baseline profile 255
A.2.2 Main profile 256
A.2.3 Extended profile 256
A.2.4 High profile 256
A.2.5 High 10 profile 257
A.2.6 High 4:2:2 profile 257
A.2.7 High 4:4:4 profile 258
A.3 Levels 258
A.3.1 Level limits common to the Baseline, Main, and Extended profiles 258
A.3.2 Level limits common to the High, High 10, High 4:2:2, and High 4:4:4 profiles 260
A.3.3 Profile-specific level limits 261
A.3.3.1 Baseline profile limits 262
A.3.3.2 Main, High, High 10, High 4:2:2, or High 4:4:4 profile limits 263
A.3.3.3 Extended Profile Limits 264
A.3.4 Effect of level limits on frame rate (informative) 266

Annex B Byte stream format 269
B.1 Byte stream NAL unit syntax and semantics 269
B.1.1 Byte stream NAL unit syntax 269
B.1.2 Byte stream NAL unit semantics 269
B.2 Byte stream NAL unit decoding process 270
B.3 Decoder byte-alignment recovery (informative) 270

Annex C Hypothetical reference decoder 272
C.1 Operation of coded picture buffer (CPB) 274
C.1.1 Timing of bitstream arrival 274
C.1.2 Timing of coded picture removal 275
C.2 Operation of the decoded picture buffer (DPB) 276
C.2.1 Decoding of gaps in frame_num and storage of "non-existing" frames 276
C.2.2 Picture decoding and output 276
C.2.3 Removal of pictures from the DPB before possible insertion of the current picture 277
C.2.4 Current decoded picture marking and storage 277
C.2.4.1 Marking and storage of a reference decoded picture into the DPB 277
C.2.4.2 Storage of a non-reference picture into the DPB 277
C.3 Bitstream conformance 277
C.4 Decoder conformance 279
C.4.1 Operation of the output order DPB 280
C.4.2 Decoding of gaps in frame_num and storage of "non-existing" pictures 280
C.4.3 Picture decoding 280
C.4.4 Removal of pictures from the DPB before possible insertion of the current picture 280
C.4.5 Current decoded picture marking and storage 281
C.4.5.1 Storage and marking of a reference decoded picture into the DPB 281
C.4.5.2 Storage and marking of a non-reference decoded picture into the DPB 281
C.4.5.3 "Bumping" process 281

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

Annex D Supplemental enhancement information **283**
D.1 SEI payload syntax 284
D.1.1 Buffering period SEI message syntax 285
D.1.2 Picture timing SEI message syntax 285
D.1.3 Pan-scan rectangle SEI message syntax 286
D.1.4 Filler payload SEI message syntax 287
D.1.5 User data registered by ITU-T Recommendation T.35 SEI message syntax 287
D.1.6 User data unregistered SEI message syntax 287
D.1.7 Recovery point SEI message syntax 287
D.1.8 Decoded reference picture marking repetition SEI message syntax 288
D.1.9 Spare picture SEI message syntax 288
D.1.10 Scene information SEI message syntax 289
D.1.11 Sub-sequence information SEI message syntax 289
D.1.12 Sub-sequence layer characteristics SEI message syntax 289
D.1.13 Sub-sequence characteristics SEI message syntax 290
D.1.14 Full-frame freeze SEI message syntax 290
D.1.15 Full-frame freeze release SEI message syntax 290
D.1.16 Full-frame snapshot SEI message syntax 290
D.1.17 Progressive refinement segment start SEI message syntax 291
D.1.18 Progressive refinement segment end SEI message syntax 291
D.1.19 Motion-constrained slice group set SEI message syntax 291
D.1.20 Film grain characteristics SEI message syntax 292
D.1.21 Deblocking filter display preference SEI message syntax 292
D.1.22 Stereo video information SEI message syntax 293
D.1.23 Reserved SEI message syntax 293
D.2 SEI payload semantics 293
D.2.1 Buffering period SEI message semantics 293
D.2.2 Picture timing SEI message semantics 294
D.2.3 Pan-scan rectangle SEI message semantics 297
D.2.4 Filler payload SEI message semantics 299
D.2.5 User data registered by ITU-T Recommendation T.35 SEI message semantics 299
D.2.6 User data unregistered SEI message semantics 299
D.2.7 Recovery point SEI message semantics 299
D.2.8 Decoded reference picture marking repetition SEI message semantics 301
D.2.9 Spare picture SEI message semantics 301
D.2.10 Scene information SEI message semantics 303
D.2.11 Sub-sequence information SEI message semantics 304
D.2.12 Sub-sequence layer characteristics SEI message semantics 306
D.2.13 Sub-sequence characteristics SEI message semantics 307
D.2.14 Full-frame freeze SEI message semantics 308
D.2.15 Full-frame freeze release SEI message semantics 308
D.2.16 Full-frame snapshot SEI message semantics 309
D.2.17 Progressive refinement segment start SEI message semantics 309
D.2.18 Progressive refinement segment end SEI message semantics 309
D.2.19 Motion-constrained slice group set SEI message semantics 310
D.2.20 Film grain characteristics SEI message semantics 310
D.2.21 Deblocking filter display preference SEI message semantics 316
D.2.22 Stereo video information SEI message semantics 318
D.2.23 Reserved SEI message semantics 318
Annex E Video usability information **319**
E.1 VUI syntax 320
E.1.1 VUI parameters syntax 320
E.1.2 HRD parameters syntax 321
E.2 VUI semantics 321
E.2.1 VUI parameters semantics 321
E.2.2 HRD parameters semantics 332
Annex F Patent Rights **334**

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

LIST OF FIGURES

Figure 6-1 – Nominal vertical and horizontal locations of 4:2:0 luma and chroma samples in a frame 16
Figure 6-2 – Nominal vertical and horizontal sampling locations of 4:2:0 samples in top and bottom fields. 17
Figure 6-3 – Nominal vertical and horizontal locations of 4:2:2 luma and chroma samples in a frame 18
Figure 6-4 – Nominal vertical and horizontal sampling locations of 4:2:2 samples top and bottom fields 18
Figure 6-5 – Nominal vertical and horizontal locations of 4:4:4 luma and chroma samples in a frame 19
Figure 6-6 – Nominal vertical and horizontal sampling locations of 4:4:4 samples top and bottom fields 19
Figure 6-7 – A picture with 11 by 9 macroblocks that is partitioned into two slices 20
Figure 6-8 – Partitioning of the decoded frame into macroblock pairs 20
Figure 6-9 – Macroblock partitions, sub-macroblock partitions, macroblock partition scans, and sub-macroblock partition scans 22
Figure 6-10 – Scan for 4x4 luma blocks 23
Figure 6-11 – Scan for 8x8 luma blocks 23
Figure 6-12 – Neighbouring macroblocks for a given macroblock 24
Figure 6-13 – Neighbouring macroblocks for a given macroblock in MBAFF frames 25
Figure 6-14 – Determination of the neighbouring macroblock, blocks, and partitions (informative) 26
Figure 7-1 – 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 18, inclusive, or in the range of 20 to 31, inclusive. 62
Figure 8-1 – Intra_4x4 prediction mode directions (informative) 120
Figure 8-2 –Example for temporal direct-mode motion vector inference (informative) 148
Figure 8-3 – Directional segmentation prediction (informative) 149
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 155
Figure 8-5 – Fractional sample position dependent variables in chroma interpolation and surrounding integer position samples A, B, C, and D 157
Figure 8-6 – Assignment of the indices of dcY to luma4x4BlkIdx 163
Figure 8-7 – Assignment of the indices of dcC to chroma4x4BlkIdx: (a) chroma_format_idc equal to 1, (b) chroma_format_idc equal to 2, (c) chroma_format_idc equal to 3 165
Figure 8-8 – 4x4 block scans. (a) Zig-zag scan. (b) Field scan (informative) 166
Figure 8-9 – 8x8 block scans. (a) 8x8 zig-zag scan. (b) 8x8 field scan (informative) 167
Figure 8-10 – Boundaries in a macroblock to be filtered 185
Figure 8-11 – Convention for describing samples across a 4x4 block horizontal or vertical boundary 189
Figure 9-1 – Illustration of CABAC parsing process for a syntax element SE (informative) 210
Figure 9-2 – Overview of the arithmetic decoding process for a single bin (informative) 242
Figure 9-3 – Flowchart for decoding a decision 244
Figure 9-4 – Flowchart of renormalization 246
Figure 9-5 – Flowchart of bypass decoding process 247
Figure 9-6 – Flowchart of decoding a decision before termination 248
Figure 9-7 – Flowchart for encoding a decision 249

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

Figure 9-8 – Flowchart of renormalization in the encoder 250
Figure 9-9 – Flowchart of PutBit(B) 251
Figure 9-10 – Flowchart of encoding bypass 252
Figure 9-11 – Flowchart of encoding a decision before termination 253
Figure 9-12 – Flowchart of flushing at termination 253
Figure C-1 – Structure of byte streams and NAL unit streams for HRD conformance checks 272
Figure C-2 – HRD buffer model 273
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 328

LIST OF TABLES

Table 6-1 –SubWidthC, and SubHeightC values derived from chroma_format_idc 15
Table 6-2 – Specification of input and output assignments for subclauses 6.4.8.1 to 6.4.8.5 26
Table 6-3 – Specification of mbAddrN 30
Table 6-4 - Specification of mbAddrN and yM 32
Table 7-1 – NAL unit type codes 57
Table 7-2 – Assignment of mnemonic names to scaling list indices and specification of fall-back rule 65
Table 7-3 – Specification of default scaling lists Default_4x4_Intra and Default_4x4_Inter 66
Table 7-4 – Specification of default scaling lists Default_8x8_Intra and Default_8x8_Inter 66
Table 7-5 – Meaning of primary_pic_type 73
Table 7-6 – Name association to slice_type 76
Table 7-7 – Reordering_of_pic_nums_idc operations for reordering of reference picture lists 81
Table 7-8 – Interpretation of adaptive_ref_pic_marking_mode_flag 83
Table 7-9 – Memory management control operation (memory_management_control_operation) values 84
Table 7-10 – Allowed collective macroblock types for slice_type 86
Table 7-11 – Macroblock types for I slices 88
Table 7-12 – Macroblock type with value 0 for SI slices 89
Table 7-13 – Macroblock type values 0 to 4 for P and SP slices 90
Table 7-14 – Macroblock type values 0 to 22 for B slices 91
Table 7-15 – Specification of CodedBlockPatternChroma values 93
Table 7-16 – Relationship between intra_chroma_pred_mode and spatial prediction modes 93
Table 7-17 – Sub-macroblock types in P macroblocks 94
Table 7-18 – Sub-macroblock types in B macroblocks 95
Table 8-1 – Refined slice group map type 104
Table 8-2 – Specification of Intra4x4PredMode[luma4x4BlkIdx] and associated names 120
Table 8-3 – Specification of Intra8x8PredMode[luma8x8BlkIdx] and associated names 126
Table 8-4 – Specification of Intra16x16PredMode and associated names 132
Table 8-5 – Specification of Intra chroma prediction modes and associated names 134

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

Table 8-6 – Specification of the variable colPic 142
Table 8-7 – Specification of PicCodingStruct(X) 142
Table 8-8 – Specification of mbAddrCol, yM, and vertMvScale 143
Table 8-9 – Assignment of prediction utilization flags 145
Table 8-10 – Derivation of the vertical component of the chroma vector in field coding mode 151
Table 8-11 – Differential full-sample luma locations 155
Table 8-12 – Assignment of the luma prediction sample predPartLX$_L$[x_L, y_L] 157
Table 8-13 – Specification of mapping of idx to c_{ij} for zig-zag and field scan 166
Table 8-14 – Specification of mapping of idx to c_{ij} for 8x8 zig-zag and 8x8 field scan 168
Table 8-15 – Specification of QP_C as a function of qP_I 169
Table 8-16 – Derivation of offset dependent threshold variables α' and β' from indexA and indexB 192
Table 8-17 – Value of variable t'_{C0} as a function of indexA and bS 194
Table 9-1 – Bit strings with “prefix” and “suffix” bits and assignment to codeNum ranges (informative) 196
Table 9-2 – Exp-Golomb bit strings and codeNum in explicit form and used as ue(v) (informative) 196
Table 9-3 – Assignment of syntax element to codeNum for signed Exp-Golomb coded syntax elements se(v) 197
Table 9-4 – Assignment of codeNum to values of coded_block_pattern for macroblock prediction modes 198
Table 9-5 – coeff_token mapping to TotalCoeff(coeff_token) and TrailingOnes(coeff_token) 203
Table 9-6 – Codeword table for level_prefix (informative) 206
Table 9-7 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 1 to 7 207
Table 9-8 – total_zeros tables for 4x4 blocks with TotalCoeff(coeff_token) 8 to 15 208
Table 9-9 – total_zeros tables for chroma DC 2x2 and 2x4 blocks 208
Table 9-10 – Tables for run_before 209
Table 9-11 – Association of ctxIdx and syntax elements for each slice type in the initialisation process 211
Table 9-12 – Values of variables m and n for ctxIdx from 0 to 10 212
Table 9-13 – Values of variables m and n for ctxIdx from 11 to 23 213
Table 9-14 – Values of variables m and n for ctxIdx from 24 to 39 213
Table 9-15 – Values of variables m and n for ctxIdx from 40 to 53 213
Table 9-16 – Values of variables m and n for ctxIdx from 54 to 59, and 399 to 401 214
Table 9-17 – Values of variables m and n for ctxIdx from 60 to 69 214
Table 9-18 – Values of variables m and n for ctxIdx from 70 to 104 215
Table 9-19 – Values of variables m and n for ctxIdx from 105 to 165 216
Table 9-20 – Values of variables m and n for ctxIdx from 166 to 226 217
Table 9-21 – Values of variables m and n for ctxIdx from 227 to 275 218
Table 9-22 – Values of variables m and n for ctxIdx from 277 to 337 219
Table 9-23 – Values of variables m and n for ctxIdx from 338 to 398 220
Table 9-24 – Values of variables m and n for ctxIdx from 402 to 459 221
Table 9-25 – Syntax elements and associated types of binarization, maxBinIdxCtx, and ctxIdxOffset 223

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Page

Table 9-26 – Bin string of the unary binarization (informative) 224
Table 9-27 – Binarization for macroblock types in I slices 227
Table 9-28 – Binarization for macroblock types in P, SP, and B slices 228
Table 9-29 – Binarization for sub-macroblock types in P, SP, and B slices 229
Table 9-30 – 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 231
Table 9-31 – Assignment of ctxIdxBlockCatOffset to ctxBlockCat for syntax elements coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, and coeff_abs_level_minus1 232
Table 9-32 – Specification of ctxIdxInc for specific values of ctxIdxOffset and binIdx 239
Table 9-33 – Specification of ctxBlockCat for the different blocks 240
Table 9-34 – Mapping of scanning position to ctxIdxInc for ctxBlockCat == 5 241
Table 9-35 – Specification of rangeTabLPS depending on pStateIdx and qCodIRangeIdx 245
Table 9-36 – State transition table 246
Table A-1 – Level limits 260
Table A-2 – Specification of cpbBrVclFactor and cpbBrNalFactor 262
Table A-3 – Baseline profile level limits 263
Table A-4 – Main, High, High 10, High 4:2:2, or High 4:4:4 profile level limits 264
Table A-5 – Extended profile level limits 265
Table A-6 – Maximum frame rates (frames per second) for some example frame sizes 266
Table D-1 – Interpretation of pic_struct 295
Table D-2 – Mapping of ct_type to source picture scan 296
Table D-3 – Definition of counting_type values 296
Table D-4 – scene_transition_type values. 303
Table D-5 – model_id values 311
Table D-6 – blending_mode_id values 312
Table E-1 – Meaning of sample aspect ratio indicator 322
Table E-2 – Meaning of video_format 323
Table E-3 – Colour primaries 324
Table E-4 – Transfer characteristics 325
Table E-5 – Matrix coefficients 327
Table E-6 – Divisor for computation of $\Delta t_{fi,dpb}(n)$ 329
Table F-1 – Organisations providing patent rights licensing notices 335

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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 Publication and versions of this International Standard

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

This International Standard has been jointly developed by ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group. It is published as technically-aligned twin text in both organizations ITU-T and ISO/IEC.

ITU-T Rec. H.264 | ISO/IEC 14496-10 version 1 refers to the first (2003) approved version of this Recommendation | International Standard.

ITU-T Rec. H.264 | ISO/IEC 14496-10 version 2 refers to the integrated text containing the corrections specified in the first technical corrigendum.

ITU-T Rec. H.264 | ISO/IEC 14496-10 version 3 refers to the integrated text containing both the first technical corrigendum (2004) and the first amendment, which is referred to as the "Fidelity range extensions".

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

ITU-T Rec. H.264 | ISO/IEC 14496-10 version 4 (the current specification) refers to the integrated text containing the first technical corrigendum (2004), the first amendment (the "Fidelity range extensions"), and an additional technical corrigendum (2005). In the ITU-T, the next published version after version 2 was version 4 (due to the completion of the drafting work for version 4 prior to the approval opportunity for a final version 3 text).

0.5 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 International Standard, 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 International Standard will facilitate video data interchange among different applications.

Considering the practicality of implementing the full syntax of this International Standard, 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.

0.6 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. With the exception of the transform bypass mode of operation for lossless coding in the High 4:4:4 profile and the I_PCM mode of operation in all profiles, the algorithm is typically 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. Encoding algorithms (not specified in this Recommendation | International Standard) may select 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.

0.6.1 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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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.

0.6.2 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.

0.6.3 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.

0.6.4 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.

0.7 How to read this specification

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

It is suggested that the reader start 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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Annex A specifies seven profiles (Baseline, Main, Extended, High, High 10, High 4:2:2 and High 4:4:4), 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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

NOTES

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

AUSTRALIAN STANDARD

Information technology — Coding of audio-visual objects —

Part 10:
Advanced Video Coding

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, *Information technology — Open Systems Interconnection — Remote Procedure Call (RPC)*
- ISO/CIE 10527:1991, *CIE standard 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 exactly one *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*.

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

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

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*.

3.5 **alpha blending**: A process not specified by this Recommendation | International Standard, in which an *auxiliary coded picture* is used in combination with a *primary coded picture* and with other data not specified by this Recommendation | International Standard in the *display process.* In an alpha blending process, the samples of an *auxiliary coded picture* are interpreted as indications of the degree of opacity (or, equivalently, the degrees of transparency) associated with the corresponding *luma* samples of the *primary coded picture.*

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

3.6 **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 less than the *macroblock address* of the first *macroblock* of some other preceding *slice* of the same *coded picture*.

3.7 **auxiliary coded picture**: A *picture* that supplements the *primary coded picture* that may be used in combination with other data not specified by this Recommendation | International Standard in the *display process.* An auxiliary coded picture has the same syntactic and semantic restrictions as a monochrome *redundant coded picture.* An auxiliary coded picture must contain the same number of *macroblocks* as the *primary coded picture*. Auxiliary coded pictures have no normative effect on the *decoding process*. See also *primary coded picture* and *redundant coded picture.*

3.8 **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*.

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

3.10 **binarization**: A set of *bin strings* for all possible values of a *syntax element*.

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

3.12 **bin string**: A string of *bins*. A bin string is an intermediate binary representation of values of *syntax elements* from the *binarization* of the *syntax element*.

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

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

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

3.16 **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*.

3.17 **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.*

3.18 **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*.

3.19 **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.

3.20 **byte-aligned**: A position in a *bitstream* is byte-aligned when the position is an integer multiple of 8 bits from the position of the first bit in the *bitstream*. A bit or *byte* or *syntax element* is said to be byte-aligned when the position at which it appears in a *bitstream* is byte-aligned.

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

3.22 **can**: A term used to refer to behaviour that is allowed, but not necessarily required.

3.23 **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.

3.24 **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.

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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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

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

3.27 **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.

3.28 **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.

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

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

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

3.32 **complementary field pair:** A collective term for a *complementary reference field pair* or a *complementary non-reference field pair*.

3.33 **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.*

3.34 **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 the frame_num *syntax element*, where the second *field* in *decoding order* is not an *IDR picture* and does not include a memory_management_control_operation *syntax element* equal to 5.

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

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

3.37 **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*.

3.38 **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.

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

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

3.41 **decoding process**: The process specified in this Recommendation | International Standard that reads a *bitstream* and derives *decoded pictures* from it.

3.42 **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.

3.43 **display process**: A process not specified in this Recommendation | International Standard having, as its input, the cropped decoded *pictures* that are the output of the *decoding process.*

3.44 **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*.

3.45 **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*.

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

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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

3.49 **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.

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

3.51 **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.*

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

3.53 **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*.

3.54 **frame macroblock**: A *macroblock* representing samples from the 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.

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

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

3.57 **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.

3.58 **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*.

3.59 **I slice**: A *slice* that is not an *SI slice* that is decoded using *prediction* only from decoded samples within the same *slice*.

3.60 **informative**: A term used to refer to content provided in this Recommendation | International Standard that is not an integral part of this Recommendation | International Standard. Informative content does not establish any mandatory requirements for conformance to this Recommendation | International Standard.

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

3.62 **instantaneous decoding refresh (IDR) picture**: A *coded picture* in which all *slices* are *I* or *SI slices* 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.

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

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

3.65 **interpretation sample value**: A possibly-altered value corresponding to a decoded sample value of an *auxiliary coded picture* that may be generated for use in the *display process*. Interpretation sample values are not used in the *decoding process* and have no normative effect on the *decoding process*.

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

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

3.68 **intra slice**: See *I slice*.

3.69 **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*.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

3.70 **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.

3.71 **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*.

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

3.73 **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*).

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

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. The symbol L is sometimes used instead of the symbol Y to avoid confusion with the symbol y as used for vertical location.

3.75 **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*.

3.76 **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*.

3.77 **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.

3.78 **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*.

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

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

3.81 **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.

3.82 **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.

3.83 **may**: A term used to refer to behaviour that is allowed, but not necessarily required. In some places where the optional nature of the described behaviour is intended to be emphasized, the phrase "may or may not" is used to provide emphasis.

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

3.85 **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*.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

3.86 **must**: A term used in expressing an observation about a requirement or an implication of a requirement that is specified elsewhere in this Recommendation | International Standard. This term is used exclusively in an *informative* context.

3.87 **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*.

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

3.89 **non-paired field**: A collective term for a *non-paired reference field* or a *non-paired non-reference field*.

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

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

3.92 **non-reference field**: A *field* coded with nal_ref_idc equal to 0.

3.93 **non-reference frame**: A *frame* coded with nal_ref_idc equal to 0.

3.94 **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*.

3.95 **note**: A term used to prefix *informative* remarks. This term is used exclusively in an *informative* context.

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

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

3.98 **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*.

3.99 **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*.

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

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

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

3.103 **picture parameter set**: A *syntax structure* containing *syntax elements* that apply to zero or more entire *coded pictures* as determined by the pic_parameter_set_id *syntax element* found in each *slice header.*

3.104 **picture order count**: A variable having a value that is non-decreasing 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".

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

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

3.107 **predictive slice**: See *P slice*.

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

3.109 **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.*

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

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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

3.113 **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.

3.114 **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.

3.115 **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*.

3.116 **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*.

3.117 **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*.

3.118 **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*.

3.119 **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*.

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

3.121 **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*.

3.122 **reference picture list**: A list of *reference pictures* that is used for *inter prediction* of a *P, B,* or *SP slice.* For the *decoding process* of a *P* or *SP slice,* there is one reference picture list. For the *decoding process* of a *B slice*, there are two reference picture lists.

3.123 **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*.

3.124 **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*.

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

3.126 **reserved**: The term reserved, when used in the clauses specifying some values of a particular *syntax element*, are for future use by ITU-T | ISO/IEC. 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.

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

3.128 **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*.

3.129 **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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

aspect ratio is expressed as *h*:*v*, where *h* is horizontal width and *v* is vertical height (in arbitrary units of spatial distance).

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

3.131 **sequence parameter set**: A *syntax structure* containing *syntax elements* that apply to zero or more entire *coded video sequences* as determined by the content of a seq_parameter_set_id *syntax element* found in the *picture parameter set* referred to by the pic_parameter_set_id *syntax element* found in each *slice header*.

3.132 **shall**: A term used to express mandatory requirements for conformance to this Recommendation | International Standard. When used to express a mandatory constraint on the values of *syntax elements* or on the results obtained by operation of the specified *decoding process*, it is the responsibility of the *encoder* to ensure that the constraint is fulfilled. When used in reference to operations performed by the *decoding process*, any *decoding process* that produces identical results to the *decoding process* described herein conforms to the *decoding process* requirements of this Recommendation | International Standard.

3.133 **should**: A term used to refer to behaviour of an implementation that is encouraged to be followed under anticipated ordinary circumstances, but is not a mandatory requirement for conformance to this Recommendation | International Standard.

3.134 **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*.

3.135 **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*.

3.136 **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*.

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

3.138 **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*.

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

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

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

3.142 **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*.

3.143 **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*.

3.144 **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.

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

3.146 **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*.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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

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

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

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

3.151 **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*.

3.152 **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*.

3.153 **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*.

3.154 **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.

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

3.156 **unspecified:** The term unspecified, when used in the clauses specifying some values of a particular *syntax element*, indicates that the values have no specified meaning in this Recommendation | International Standard and will not have a specified meaning in the future as an integral part of this Recommendation | International Standard.

3.157 **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.

3.158 **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*.

4 Abbreviations

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

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

4.3 **CBR**: Constant Bit Rate

4.4 **CPB**: Coded Picture Buffer

4.5 **DPB**: Decoded Picture Buffer

4.6 **DUT**: Decoder under test

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

4.8 **HRD**: Hypothetical Reference Decoder

4.9 **HSS**: Hypothetical Stream Scheduler

4.10 **IDR**: Instantaneous Decoding Refresh

4.11 **LSB**: Least Significant Bit

4.12 **MB**: Macroblock

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

4.14 **MSB**: Most Significant Bit

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

4.15 **NAL**: Network Abstraction Layer

4.16 **RBSP**: Raw Byte Sequence Payload

4.17 **SEI**: Supplemental Enhancement Information

4.18 **SODB**: String Of Data Bits

4.19 **UUID**: Universal Unique Identifier

4.20 **VBR**: Variable Bit Rate

4.21 **VCL**: Video Coding Layer

4.22 **VLC**: Variable Length Coding

4.23 **VUI**: Video Usability Information

5 Conventions

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.

5.1 Arithmetic operators

The following arithmetic operators are defined as follows.

+	Addition
–	Subtraction (as a two-argument operator) or negation (as a unary prefix operator)
*	Multiplication
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.
/	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.
÷	Used to denote division in mathematical equations where no truncation or rounding is intended.
$\frac{x}{y}$	Used to denote division in mathematical equations where no truncation or rounding is intended.
$\sum_{i=x}^{y} f(i)$	The summation of f(i) with i taking all integer values from x up to and including y.
x % y	Modulus. Remainder of x divided by y, defined only for integers x and y with x >= 0 and y > 0.

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

- 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

5.2 Logical operators

The following logical operators are defined as follows

x && y	Boolean logical "and" of x and y
x || y	Boolean logical "or" of x and y
!	Boolean logical "not"
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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

5.3 Relational operators

The following relational operators are defined as follows

>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
= =	Equal to
!=	Not equal to

When a relational operator is applied to a syntax element or variable that has been assigned the value "na" (not applicable), the value "na" is treated as a distinct value for the syntax element or variable. The value "na" is considered not to be equal to any other value.

5.4 Bit-wise operators

The following bit-wise operators are defined as follows

&	Bit-wise "and". When operating on integer arguments, operates on a two's complement representation of the integer value. When operating on a binary argument that contains fewer bits than another argument, the shorter argument is extended by adding more significant bits equal to 0.
|	Bit-wise "or". When operating on integer arguments, operates on a two's complement representation of the integer value. When operating on a binary argument that contains fewer bits than another argument, the shorter argument is extended by adding more significant bits equal to 0.
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 integer 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 << y	Arithmetic left shift of a two's complement integer representation of x by y binary digits. This function is defined only for positive integer values of y. Bits shifted into the LSBs as a result of the left shift have a value equal to 0.

5.5 Assignment operators

The following arithmetic operators are defined as follows

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

5.6 Range notation

The following notation is used to specify a range of values

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

5.7 Mathematical functions

The following mathematical functions are defined as follows

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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

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

$$\text{Clip1}_Y(x) = \text{Clip3}(0, (1 << \text{BitDepth}_Y) - 1 , x) \tag{5-3}$$

$$\text{Clip1}_C(x) = \text{Clip3}(0, (1 << \text{BitDepth}_C) - 1 , x) \tag{5-4}$$

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

Floor(x) the greatest integer less than or equal to x. (5-6)

$$\text{InverseRasterScan}(a, b, c, d, e) = \begin{cases} (a\%(d/b)) * b; & e == 0 \\ (a/(d/b)) * c; & e == 1 \end{cases} \tag{5-7}$$

Log2(x) returns the base-2 logarithm of x. (5-8)

Log10(x) returns the base-10 logarithm of x. (5-9)

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

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

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

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5) \tag{5-13}$$

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x >= 0 \\ -1 & ; \quad x < 0 \end{cases} \tag{5-14}$$

$$\text{Sqrt}(x) = \sqrt{x} \tag{5-15}$$

5.8 Variables, syntax elements, and tables

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.

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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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.

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

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).

A one-dimensional array is referred to as a list. A two-dimensional array is referred to as a matrix. Arrays can either be syntax elements or variables. Subscripts or square parentheses are used for the indexing of arrays. In reference to a visual depiction of a matrix, the first subscript is used as a row (vertical) index and the second subscript is used as a column (horizontal) index. The indexing order is reversed when using square parentheses rather than subscripts for indexing. Thus, an element of a matrix s at horizontal position x and vertical position y may be denoted either as s[x, y] or as s_{yx}.

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.

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

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

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.

5.9 Text description of logical operations

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

```
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:

... as follows / ... the following applies.

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

Each "If...Otherwise, if...Otherwise, ..." statement in the text is introduced with "... as follows" or "... the following applies" immediately followed by "If ... ". The last condition of the "If...Otherwise, if...Otherwise, ..." is always an "Otherwise, ...". Interleaved "If...Otherwise, if...Otherwise, ..." statements can be identified by matching "... as follows" or "... the following applies" with the ending "Otherwise, ...".

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

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

else
statement n

may be described in the following manner:

... as follows / ... the following applies.

- If all of the following conditions are true, statement 0
  - condition 0a
  - condition 0b
- Otherwise, if any of the following conditions are true, statement 1
  - condition 1a
  - condition 1b
- …
- Otherwise, statement n

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

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

may be described in the following manner:

When condition 0, statement 0

When condition 1, statement 1

5.10 Processes

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.

The assignment of variables is specified as follows.

- 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.

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.

6 Source, coded, decoded and 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.

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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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 0.

6.2 Source, decoded, and output picture formats

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

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.

The source and decoded pictures (frames or fields) are each comprised of one or more sample arrays:

- Luma (Y) only (monochrome), with or without an auxiliary array
- Luma and two Chroma (YCbCr or YCgCo), with or without an auxiliary array
- Green, Blue and Red (GBR, also known as RGB), with or without an auxiliary array
- Arrays representing other unspecified monochrome or tri-stimulus colour samplings (for example, YZX, also known as XYZ), with or without an auxiliary array

For convenience of notation and terminology in this specification, the variables and terms associated with these arrays are referred to as luma (or L or Y) and chroma, where the two chroma arrays are referred to as Cb and Cr; regardless of the actual colour representation method in use. The actual colour representation method in use can be indicated in syntax that is specified in Annex E. The (monochrome) auxiliary arrays, which may or may not be present as auxiliary pictures in a coded video sequence, are optional for decoding and can be used for such purposes as alpha blending.

The variables SubWidthC, and SubHeightC are specified in Table 6-1, depending on the chroma format sampling structure, which is specified through chroma_format_idc. An entry marked as "-" in Table 6-1 denotes an undefined value for SubWidthC or SubHeightC. Other values of chroma_format_idc, SubWidthC, and SubHeightC may be specified in the future by ITU-T | ISO/IEC.

Table 6-1 –SubWidthC, and SubHeightC values derived from chroma_format_idc

chroma_format_idc	Chroma Format	SubWidthC	SubHeightC
0	monochrome	-	-
1	4:2:0	2	2
2	4:2:2	2	1
3	4:4:4	1	1

In monochrome sampling there is only one sample array, which shall nominally be considered the 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.

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

The syntax for the luma and (when present) chroma arrays are ordered such when data for all three colour components is present, the data for the luma array is first, followed by any data for the Cb array, followed by any data for the Cr array, unless otherwise specified.

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).

The number of bits necessary for the representation of each of the samples in the luma and chroma arrays in a video sequence is in the range of 8 to 12, and the number of bits used in the luma array may differ from the number of bits used in the chroma arrays.

When the value of chroma_format_idc is equal to 1, 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).

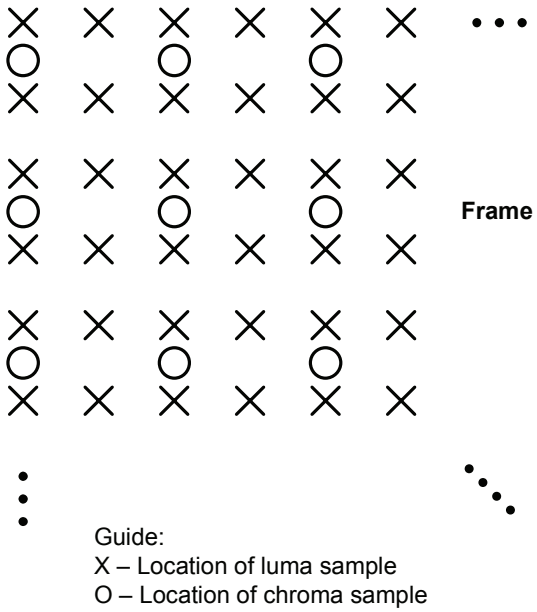

Figure 6-1 – Nominal vertical and horizontal locations of 4:2:0 luma and chroma samples in a frame

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.

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.

When the value of chroma_format_idc is equal to 1, 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).

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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

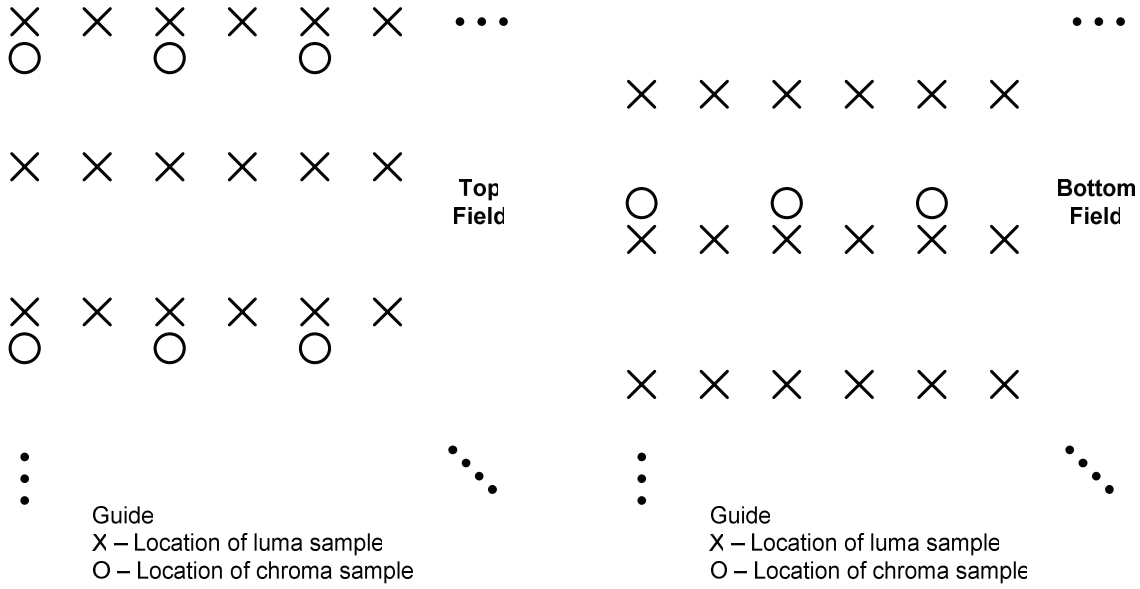

Figure 6-2 – Nominal vertical and horizontal sampling locations of 4:2:0 samples in top and bottom fields.

When the value of chroma_format_idc is equal to 2, the chroma samples are co-sited with the corresponding luma samples and the nominal locations in a frame and in fields are as shown in Figure 6-3 and Figure 6-4, respectively.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

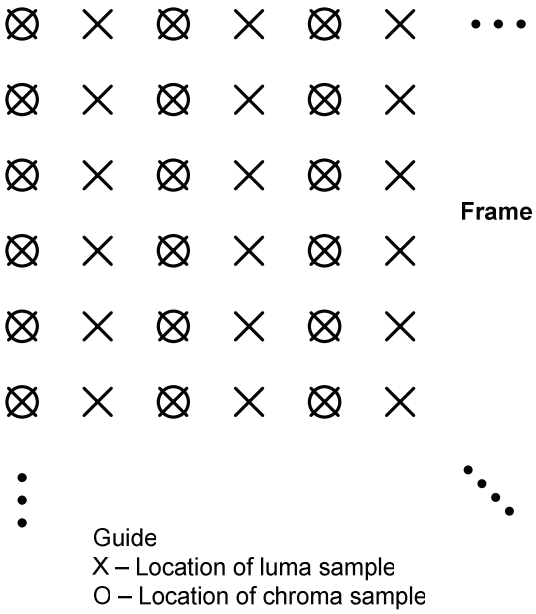

Figure 6-3 – Nominal vertical and horizontal locations of 4:2:2 luma and chroma samples in a frame

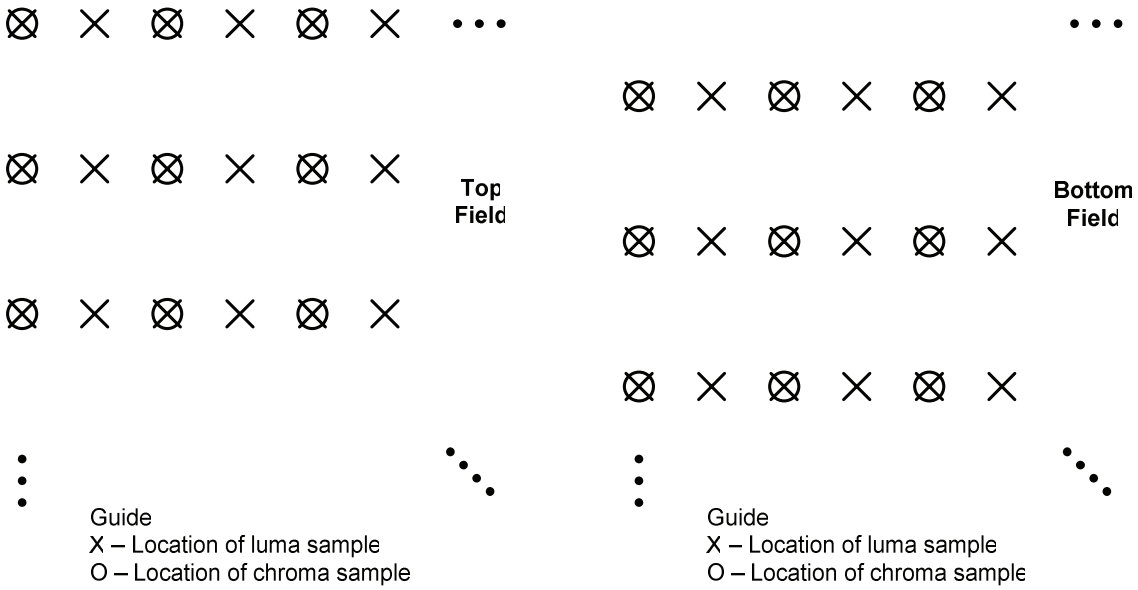

Figure 6-4 – Nominal vertical and horizontal sampling locations of 4:2:2 samples top and bottom fields

When the value of chroma_format_idc is equal to 3, all array samples are co-sited for all cases of frames and fields and the nominal locations in a frame and in fields are as shown in Figure 6-5 and Figure 6-6, respectively.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

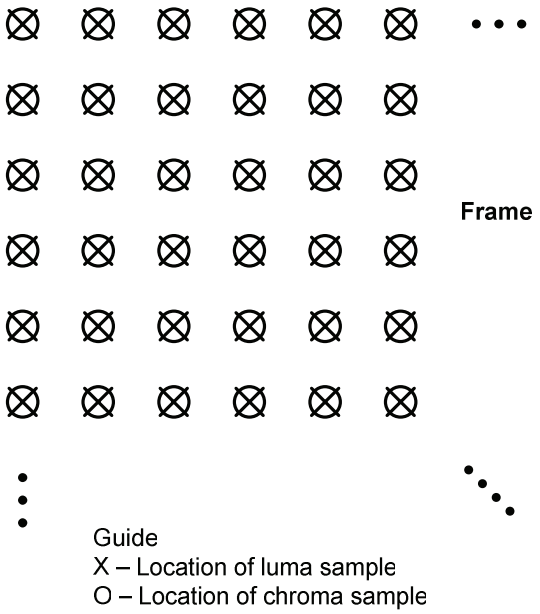

Figure 6-5 – Nominal vertical and horizontal locations of 4:4:4 luma and chroma samples in a frame

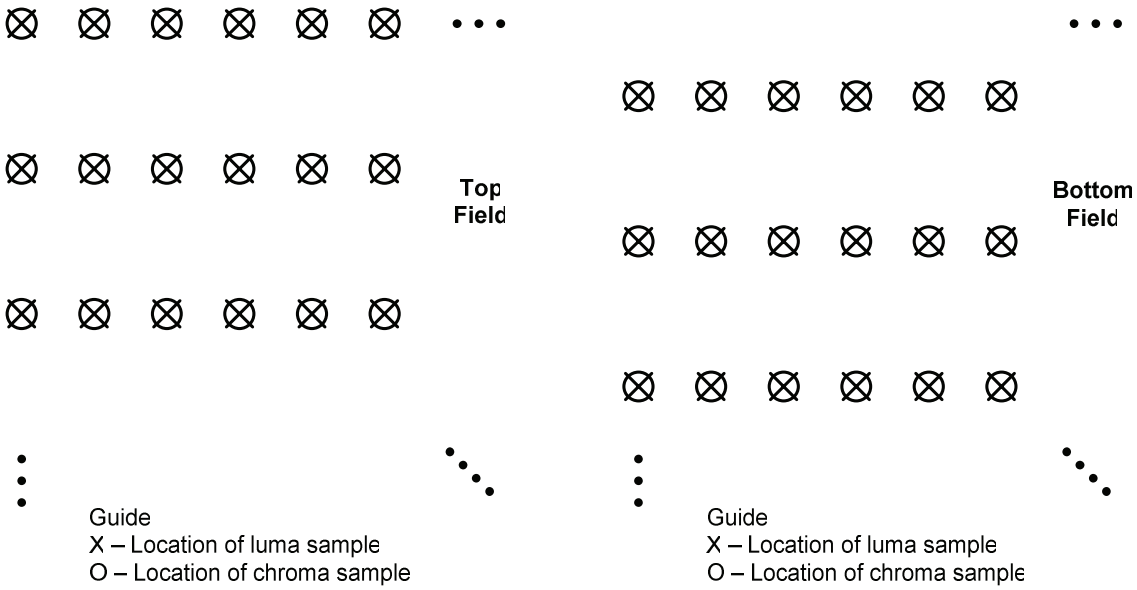

Figure 6-6 – Nominal vertical and horizontal sampling locations of 4:4:4 samples top and bottom fields

The samples are processed in units of macroblocks. The luma array for each macroblock is 16 samples in both width and height. The variables MbWidthC and MbHeightC, which specify the width and height, respectively, of the chroma arrays for each macroblock, are derived as follows.

– If chroma_format_idc is equal to 0 (monochrome), MbWidthC and MbHeightC are both equal to 0 (as no chroma arrays are specified for monochrome video).

– Otherwise, MbWidthC and MbHeightC are derived as

$$MbWidthC = 16 / SubWidthC \quad (6\text{-}1)$$

$$MbHeightC = 16 / SubHeightC \quad (6\text{-}2)$$

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

AVA Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

6.3 Spatial subdivision of pictures and slices

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.

Each macroblock is comprised of one 16x16 luma array and, when the video format is not monochrome, two corresponding 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 6-7.

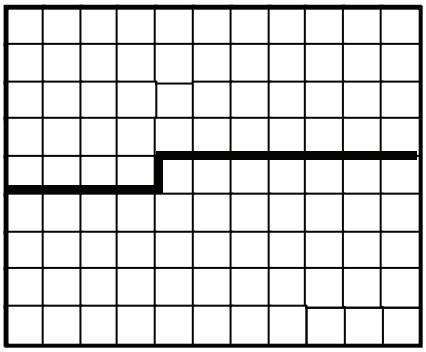

Figure 6-7 – A picture with 11 by 9 macroblocks that is partitioned into two slices

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 6-8. Each macroblock pair consists of two macroblocks.

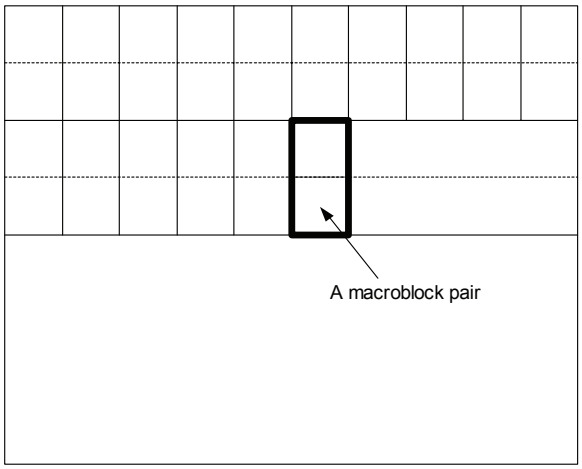

Figure 6-8 – Partitioning of the decoded frame into macroblock pairs

6.4 Inverse scanning processes and derivation processes for neighbours

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

6.4.1 Inverse macroblock scanning process

Input to this process is a macroblock address 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.

The inverse macroblock scanning process is specified as follows.

- If MbaffFrameFlag is equal to 0,

$$x = \text{InverseRasterScan}(\text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 0) \quad (6\text{-}3)$$

$$y = \text{InverseRasterScan}(\text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 1) \quad (6\text{-}4)$$

- Otherwise (MbaffFrameFlag is equal to 1), the following applies.

$$xO = \text{InverseRasterScan}(\text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 0) \quad (6\text{-}5)$$

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

Depending on the current macroblock the following applies.

– If the current macroblock is a frame macroblock

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

$$y = yO + (\text{mbAddr} \% 2) * 16 \quad (6\text{-}8)$$

– Otherwise (the current macroblock is a field macroblock),

$$x = xO \quad (6\text{-}9)$$

$$y = yO + (\text{mbAddr} \% 2) \quad (6\text{-}10)$$

6.4.2 Inverse macroblock partition and sub-macroblock partition scanning process

Macroblocks or sub-macroblocks may be partitioned, and the partitions are scanned for inter prediction as shown in Figure 6-9. 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.

The functions MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ), and SubMbPartHeight( ) describing the width and height of macroblock partitions and sub-macroblock partitions are specified in Table 7-13, Table 7-14, Table 7-17, and Table 7-18. 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 macroblock with mb_type equal to P_8x8, P_8x8ref0, or B_8x8, depending on the sub-macroblock type.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

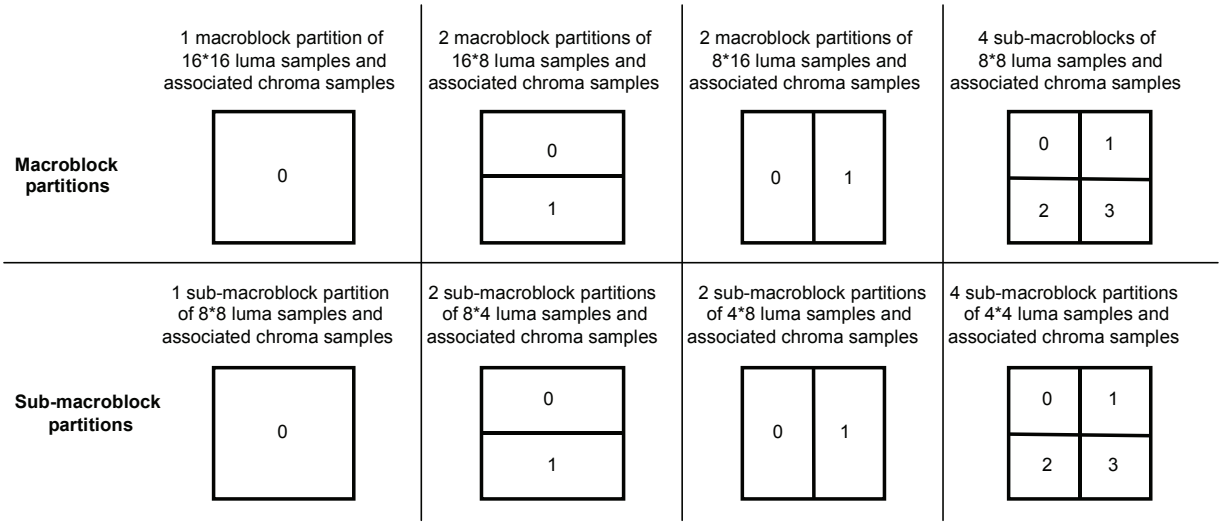

Figure 6-9 – Macroblock partitions, sub-macroblock partitions, macroblock partition scans, and sub-macroblock partition scans

6.4.2.1 Inverse macroblock partition scanning process

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

The inverse macroblock partition scanning process is specified by

$$x = \text{InverseRasterScan}(\text{mbPartIdx}, \text{MbPartWidth}(\text{mb_type}), \text{MbPartHeight}(\text{mb_type}), 16, 0) \quad (6\text{-}11)$$

$$y = \text{InverseRasterScan}(\text{mbPartIdx}, \text{MbPartWidth}(\text{mb_type}), \text{MbPartHeight}(\text{mb_type}), 16, 1) \quad (6\text{-}12)$$

6.4.2.2 Inverse sub-macroblock partition scanning process

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

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.

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

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

$$x = \text{InverseRasterScan}(\text{subMbPartIdx}, \text{SubMbPartWidth}(\text{sub_mb_type}[\text{mbPartIdx}]), \text{SubMbPartHeight}(\text{sub_mb_type}[\text{mbPartIdx}]), 8, 0) \quad (6\text{-}13)$$

$$y = \text{InverseRasterScan}(\text{subMbPartIdx}, \text{SubMbPartWidth}(\text{sub_mb_type}[\text{mbPartIdx}]), \text{SubMbPartHeight}(\text{sub_mb_type}[\text{mbPartIdx}]), 8, 1) \quad (6\text{-}14)$$

- Otherwise,

$$x = \text{InverseRasterScan}(\text{subMbPartIdx}, 4, 4, 8, 0) \quad (6\text{-}15)$$

$$y = \text{InverseRasterScan}(\text{subMbPartIdx}, 4, 4, 8, 1) \quad (6\text{-}16)$$

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

6.4.3 Inverse 4x4 luma block scanning process

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

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.

Figure 6-10 shows the scan for the 4x4 luma blocks.

0	1	4	5
2	3	6	7
8	9	12	13
10	11	14	15

Figure 6-10 – Scan for 4x4 luma blocks

The inverse 4x4 luma block scanning process is specified by

x = InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 0) + InverseRasterScan(luma4x4BlkIdx % 4, 4, 4, 8, 0) (6-17)

y = InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 1) + InverseRasterScan(luma4x4BlkIdx % 4, 4, 4, 8, 1) (6-18)

6.4.4 Inverse 8x8 luma block scanning process

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

Output of this process is the location (x, y) of the upper-left luma sample for the 8x8 luma block with index luma8x8BlkIdx relative to the upper-left luma sample of the macroblock.

Figure 6-11 shows the scan for the 8x8 luma blocks.

0	1
2	3

Figure 6-11 – Scan for 8x8 luma blocks

The inverse 8x8 luma block scanning process is specified by

x = InverseRasterScan(luma8x8BlkIdx, 8, 8, 16, 0) (6-19)

y = InverseRasterScan(luma8x8BlkIdx, 8, 8, 16, 1) (6-20)

6.4.5 Derivation process of the availability for macroblock addresses

Input to this process is a macroblock address mbAddr.

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

NOTE – The meaning of availability is determined when this process is invoked.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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:

- mbAddr < 0
- mbAddr > CurrMbAddr
- the macroblock with address mbAddr belongs to a different slice than the macroblock with address CurrMbAddr

6.4.6 Derivation process for neighbouring macroblock addresses and their availability

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

The outputs of this process are

- mbAddrA: the address and availability status of the macroblock to the left of the current macroblock.
- mbAddrB: the address and availability status of the macroblock above the current macroblock.
- mbAddrC: the address and availability status of the macroblock above-right of the current macroblock.
- mbAddrD: the address and availability status of the macroblock above-left of the current macroblock.

Figure 6-12 shows the relative spatial locations of the macroblocks with mbAddrA, mbAddrB, mbAddrC, and mbAddrD relative to the current macroblock with CurrMbAddr.

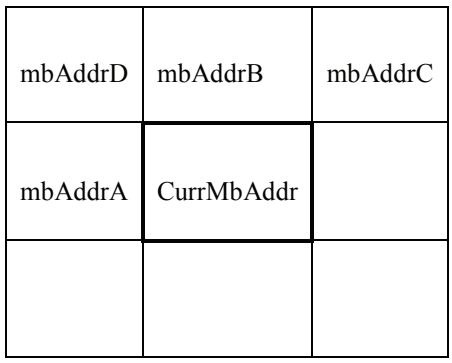

Figure 6-12 – Neighbouring macroblocks for a given macroblock

Input to the process in subclause 6.4.5 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.

Input to the process in subclause 6.4.5 is mbAddrB = CurrMbAddr – PicWidthInMbs and the output is whether the macroblock mbAddrB is available.

Input to the process in subclause 6.4.5 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.

Input to the process in subclause 6.4.5 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.

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

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

The outputs of this process are

- mbAddrA: the address and availability status of the top macroblock of the macroblock pair to the left of the current macroblock pair.
- mbAddrB: the address and availability status of the top macroblock of the macroblock pair above the current macroblock pair.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

- mbAddrC: the address and availability status of the top macroblock of the macroblock pair above-right of the current macroblock pair.
- mbAddrD: the address and availability status of the top macroblock of the macroblock pair above-left of the current macroblock pair.

Figure 6-13 shows the relative spatial locations of the macroblocks with mbAddrA, mbAddrB, mbAddrC, and mbAddrD relative to the current macroblock with CurrMbAddr.

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

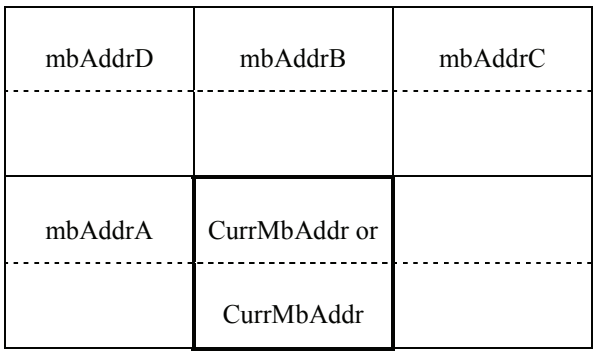

Figure 6-13 – Neighbouring macroblocks for a given macroblock in MBAFF frames

Input to the process in subclause 6.4.5 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.

Input to the process in subclause 6.4.5 is mbAddrB =2 * (CurrMbAddr / 2 – PicWidthInMbs) and the output is whether the macroblock mbAddrB is available.

Input to the process in subclause 6.4.5 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.

Input to the process in subclause 6.4.5 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.

6.4.8 Derivation processes for neighbouring macroblocks, blocks, and partitions

Subclause 6.4.8.1 specifies the derivation process for neighbouring macroblocks.

Subclause 6.4.8.2 specifies the derivation process for neighbouring 8x8 luma blocks.

Subclause 6.4.8.3 specifies the derivation process for neighbouring 4x4 luma blocks.

Subclause 6.4.8.4 specifies the derivation process for neighbouring 4x4 chroma blocks.

Subclause 6.4.8.5 specifies the derivation process for neighbouring partitions.

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 6.4.8.1 to 6.4.8.5. The variable predPartWidth is specified when Table 6-2 is referred to.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Table 6-2 – Specification of input and output assignments for subclauses 6.4.8.1 to 6.4.8.5

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

Figure 6-14 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.

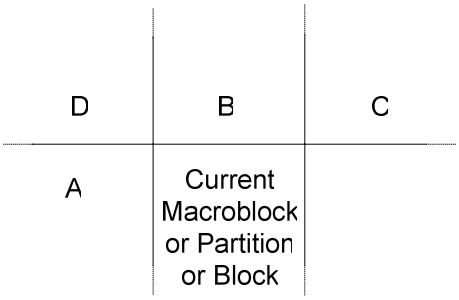

Figure 6-14 – Determination of the neighbouring macroblock, blocks, and partitions (informative)

6.4.8.1 Derivation process for neighbouring macroblocks

Outputs of this process are

- mbAddrA: the address of the macroblock to the left of the current macroblock and its availability status and
- mbAddrB: the address of the macroblock above the current macroblock and its availability status.

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

- The difference of luma location (xD, yD) is set according to Table 6-2.
- The derivation process for neighbouring locations as specified in subclause 6.4.9 is invoked for luma locations with (xN, yN) equal to (xD, yD), and the output is assigned to mbAddrN.

6.4.8.2 Derivation process for neighbouring 8x8 luma block

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

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

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Outputs of this process are

- mbAddrA: either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock and its availability status,
- luma8x8BlkIdxA: the index of the 8x8 luma block to the left of the 8x8 block with index luma8x8BlkIdx and its availability status,
- mbAddrB: either equal to CurrMbAddr or the address of the macroblock above the current macroblock and its availability status,
- luma8x8BlkIdxB: the index of the 8x8 luma block above the 8x8 block with index luma8x8BlkIdx and its availability status.

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

- The difference of luma location (xD, yD) is set according to Table 6-2.
- The luma location (xN, yN) is specified by

$$xN = (luma8x8BlkIdx \% 2) * 8 + xD \quad (6\text{-}21)$$

$$yN = (luma8x8BlkIdx / 2) * 8 + yD \quad (6\text{-}22)$$

- The derivation process for neighbouring locations as specified in subclause 6.4.9 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- The variable luma8x8BlkIdxN is derived as follows.
  - If mbAddrN is not available, luma8x8BlkIdxN is marked as not available.
  - Otherwise (mbAddrN is available), the 8x8 luma block in the macroblock mbAddrN covering the luma location (xW, yW) shall be assigned to luma8x8BlkIdxN.

6.4.8.3 Derivation process for neighbouring 4x4 luma blocks

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

Outputs of this process are

- mbAddrA: either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock and its availability status,
- luma4x4BlkIdxA: the index of the 4x4 luma block to the left of the 4x4 block with index luma4x4BlkIdx and its availability status,
- mbAddrB: either equal to CurrMbAddr or the address of the macroblock above the current macroblock and its availability status,
- luma4x4BlkIdxB: the index of the 4x4 luma block above the 4x4 block with index luma4x4BlkIdx and its availability status.

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

- The difference of luma location (xD, yD) is set according to Table 6-2.
- 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.
- The luma location (xN, yN) is specified by

$$xN = x + xD \quad (6\text{-}23)$$

$$yN = y + yD \quad (6\text{-}24)$$

- The derivation process for neighbouring locations as specified in subclause 6.4.9 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- The variable luma4x4BlkIdxN is derived as follows.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

- If mbAddrN is not available, luma4x4BlkIdxN is marked as not available.
- Otherwise (mbAddrN is available), the 4x4 luma block in the macroblock mbAddrN covering the luma location (xW, yW) shall be assigned to luma4x4BlkIdxN.

6.4.8.4 Derivation process for neighbouring 4x4 chroma blocks

Input to this process is a 4x4 chroma block index chroma4x4BlkIdx.

Outputs of this process are

- mbAddrA (either equal to CurrMbAddr or the address of the macroblock to the left of the current macroblock) and its availability status,
- chroma4x4BlkIdxA (the index of the 4x4 chroma block to the left of the 4x4 chroma block with index chroma4x4BlkIdx) and its availability status,
- mbAddrB (either equal to CurrMbAddr or the address of the macroblock above the current macroblock) and its availability status,
- chroma4x4BlkIdxB (the index of the 4x4 chroma block above the 4x4 chroma block with index chroma4x4BlkIdx) and its availability status.

mbAddrN and chroma4x4BlkIdxN (with N being A or B) are derived as follows.

- The difference of chroma location (xD, yD) is set according to Table 6-2.
- Depending on chroma_format_idc, the position (x, y) of the upper-left sample of the 4x4 chroma block with index chroma4x4BlkIdx is derived as follows
  - If chroma_format_idc is equal to 1 or 2, the following applies

    x = InverseRasterScan(chroma4x4BlkIdx, 4, 4, 8, 0) (6-25)

    y = InverseRasterScan(chroma4x4BlkIdx, 4, 4, 8, 1) (6-26)

  - Otherwise (chroma_format_idc is equal to 3), the following applies

    x = InverseRasterScan(chroma4x4BlkIdx / 4, 8, 8, 16, 0) +
    InverseRasterScan(chroma4x4BlkIdx % 4, 4, 4, 8, 0) (6-27)

    y = InverseRasterScan(chroma4x4BlkIdx / 4, 8, 8, 16, 1) +
    InverseRasterScan(chroma4x4BlkIdx % 4, 4, 4, 8, 1) (6-28)

- The chroma location (xN, yN) is specified by

  xN = x + xD (6-29)

  yN = y + yD (6-30)

- The derivation process for neighbouring locations as specified in subclause 6.4.9 is invoked for chroma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- The variable chroma4x4BlkIdxN is derived as follows.
  - If mbAddrN is not available, chroma4x4BlkIdxN is marked as not available.
  - Otherwise (mbAddrN is available), the 4x4 chroma block in the macroblock mbAddrN covering the chroma location (xW, yW) is assigned to chroma4x4BlkIdxN.

6.4.8.5 Derivation process for neighbouring partitions

Inputs to this process are

- a macroblock partition index mbPartIdx
- a current sub-macroblock type currSubMbType
- a sub-macroblock partition index subMbPartIdx

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

Outputs of this process are

- 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,
- 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,
- 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,
- 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.

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

- 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.
- The location of the upper-left luma sample inside a macroblock partition (xS, yS) is derived as follows.
  - 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.
  - Otherwise, (xS, yS) are set to (0, 0).
- The variable predPartWidth in Table 6-2 is specified as follows.
  - If mb_type is equal to P_Skip, B_Skip, or B_Direct_16x16, predPartWidth = 16.
  - Otherwise, if mb_type is equal to B_8x8, the following applies.
    - If currSubMbType is equal to B_Direct_8x8, predPartWidth = 16.

      NOTE – When currSubMbType is equal to B_Direct_8x8 and direct_spatial_mv_pred_flag is equal to 1, the predicted motion vector is the predicted motion vector for the complete macroblock.
    - Otherwise, predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]).
  - Otherwise, if mb_type is equal to P_8x8 or P_8x8ref0, predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]).
  - Otherwise, predPartWidth = MbPartWidth(mb_type).
- The difference of luma location (xD, yD) is set according to Table 6-2.
- The neighbouring luma location (xN, yN) is specified by

$$xN = x + xS + xD \tag{6-31}$$

$$yN = y + yS + yD \tag{6-32}$$

- The derivation process for neighbouring locations as specified in subclause 6.4.9 is invoked for luma locations with (xN, yN) as the input and the output is assigned to mbAddrN and (xW, yW).
- Depending on mbAddrN, the following applies.
  - If mbAddrN is not available, the macroblock or sub-macroblock partition mbAddrN\mbPartIdxN\subMbPartIdxN is marked as not available.
  - Otherwise (mbAddrN is available), the following applies.
    - 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.
    - 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.

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006

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.

6.4.9 Derivation process for neighbouring locations

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

Outputs of this process are

- mbAddrN: either equal to CurrMbAddr or to the address of neighbouring macroblock that contains (xN, yN) and its availability status,
- (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).

Let maxW and maxH be variables specifying maximum values of the location components xN, xW, and yN, yW, respectively. maxW and maxH are derived as follows.

- If this process is invoked for neighbouring luma locations,

$$maxW = maxH = 16 \quad (6\text{-}33)$$

- Otherwise (this process is invoked for neighbouring chroma locations),

$$maxW = MbWidthC \quad (6\text{-}34)$$

$$maxH = MbHeightC \quad (6\text{-}35)$$

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

- If MbaffFrameFlag is equal to 0, the specification for neighbouring locations in fields and non-MBAFF frames as described in subclause 6.4.9.1 is applied.
- Otherwise (MbaffFrameFlag is equal to 1), the specification for neighbouring locations in MBAFF frames as described in subclause 6.4.9.2 is applied.

6.4.9.1 Specification for neighbouring locations in fields and non-MBAFF frames

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

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.

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

Table 6-3 – Specification of mbAddrN

xN	yN	mbAddrN
< 0	< 0	mbAddrD
< 0	0 .. maxH - 1	mbAddrA
0 .. maxW - 1	< 0	mbAddrB
0 .. maxW - 1	0 .. maxH - 1	CurrMbAddr
> maxW - 1	< 0	mbAddrC
> maxW - 1	0 .. maxH - 1	not available
	> maxH - 1	not available

Accessed by MONASH UNIVERSITY LIBRARY on 15 Jun 2015 (Document currency not guaranteed when printed)

Documents
Document Sharing Hub

Full document details: AS ISO IEC 14496.10-2006