INTERNATIONAL STANDARD

ISO/IEC 23008-2

Second edition
2015-05-01

AMENDMENT 1
2015-10-01

Information technology — High efficiency coding and media delivery in heterogeneous environments —

Part 2: High efficiency video coding

AMENDMENT 1: 3D video extensions

Technologies de l'information — Codage à haute efficacité et livraison des medias dans des environnements hétérogènes —

Partie 2: Codage vidéo à haute efficacité

AMENDEMENT 1: Extensions vidéo 3D

Reference number
ISO/IEC 23008-2:2015/Amd.1:2015(E)

Documents
Document Sharing Hub

© ISO/IEC 2015

Full document details: ISO_IEC 23008-2_2015_Amd 1_2015

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2015, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

© ISO/IEC 2015 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO_IEC 23008-2_2015_Amd 1_2015

CONTENTS

Page

Annex I 3D high efficiency video coding 5
I.1 Scope 5
I.2 Normative references 5
I.3 Definitions 5
I.4 Abbreviations 5
I.5 Conventions 5
I.6 Bitstream and picture formats, partitionings, scanning processes, and neighbouring relationships 6
I.6.1 Bitstream formats 6
I.6.2 Source, decoded, and output picture formats 6
I.6.3 Partitioning of pictures, slices, slice segments, tiles, coding tree units, and coding tree blocks 6
I.6.4 Availability processes 6
I.6.5 Scanning processes 6
I.6.6 Derivation process for a wedgelet partition pattern table 6
I.6.6.1 Wedgelet partition pattern generation process 7
I.6.6.2 Wedgelet partition pattern table insertion process 7
I.7 Syntax and semantics 8
I.7.1 Method of specifying syntax in tabular form 8
I.7.2 Specification of syntax functions, categories, and descriptors 8
I.7.3 Syntax in tabular form 8
I.7.3.1 NAL unit syntax 8
I.7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax 8
I.7.3.3 Profile, tier and level syntax 13
I.7.3.4 Scaling list data syntax 13
I.7.3.5 Supplemental enhancement information message syntax 13
I.7.3.6 Slice segment header syntax 13
I.7.3.7 Short-term reference picture set syntax 16
I.7.3.8 Slice segment data syntax 17
I.7.4 Semantics 21
I.7.4.1 General 21
I.7.4.2 NAL unit semantics 21
I.7.4.3 Raw byte sequence payloads, trailing bits, and byte alignment semantics 21
I.7.4.4 Profile, tier and level semantics 27
I.7.4.5 Scaling list data semantics 27
I.7.4.6 Supplemental enhancement information message semantics 27
I.7.4.7 Slice segment header semantics 28
I.7.4.8 Short-term reference picture set semantics 31
I.7.4.9 Slice segment data semantics 31
I.8 Decoding process 36
I.8.1 General decoding process 36
I.8.1.1 General 36
I.8.1.2 Decoding process for a coded picture with nuh_layer_id greater than 0 36
I.8.2 NAL unit decoding process 36
I.8.3 Slice decoding process 36
I.8.3.1 Derivation process for the candidate picture list for disparity vector derivation 36
I.8.3.2 Derivation process for the default reference view order index for disparity derivation 37
I.8.3.3 Derivation process for a depth look-up table 37
I.8.3.4 Derivation process for the alternative target reference index for temporal motion vector prediction in merge mode 38
I.8.3.5 Derivation process for the target reference index for residual prediction 38
I.8.4 Decoding process for coding units coded in intra prediction mode 38
I.8.4.1 General decoding process for coding units coded in intra prediction mode 38
I.8.4.2 Derivation process for luma intra prediction mode 39
I.8.4.3 Derivation process for chroma intra prediction mode 40
I.8.4.4 Decoding process for intra blocks 40
I.8.5 Decoding process for coding units coded in inter prediction mode 46

Documents
Document Sharing Hub

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

I.8.5.1 General decoding process for coding units coded in inter prediction mode ... 46
I.8.5.2 Inter prediction process ... 47
I.8.5.3 Decoding process for prediction units in inter prediction mode ... 47
I.8.5.4 Decoding process for the residual signal of coding units coded in inter prediction mode ... 75
I.8.5.5 Derivation process for a disparity vector for texture layers ... 76
I.8.5.6 Derivation process for a disparity vector for depth layers ... 78
I.8.5.7 Derivation process for a depth or disparity sample array from a depth picture ... 78
I.8.6 Scaling, transformation and array construction process prior to deblocking filter process ... 80
I.8.7 In-loop filter process ... 80
I.9 Parsing process ... 80
I.9.1 General ... 80
I.9.2 Parsing process for 0-th order Exp-Golomb codes ... 80
I.9.3 CABAC parsing process for slice segment data ... 80
I.9.3.1 General ... 80
I.9.3.2 Initialization process ... 80
I.9.3.3 Binarization process ... 83
I.9.3.4 Decoding process flow ... 85
I.9.3.5 Arithmetic encoding process (informative) ... 87
I.10 Specification of bitstream subsets ... 87
I.11 Profiles, tiers, and levels ... 87
I.11.1 Profiles ... 87
I.11.1.1 3D Main profile ... 87
I.11.2 Tiers and levels ... 89
I.11.3 Decoder capabilities ... 89
I.12 Byte stream format ... 89
I.13 Hypothetical reference decoder ... 89
I.14 Supplemental enhancement information ... 89
I.14.1 General ... 89
I.14.2 SEI payload syntax ... 89
I.14.2.1 General SEI payload syntax ... 89
I.14.2.2 Annex D, Annex F, and Annex G SEI message syntax for 3D high efficiency video coding ... 89
I.14.2.3 Alternative depth information SEI message syntax ... 89
I.14.3 SEI payload semantics ... 91
I.14.3.1 General SEI payload semantics ... 91
I.14.3.2 Annex D, Annex F, and Annex G SEI message semantics for 3D high efficiency video coding ... 91
I.14.3.3 Alternative depth information SEI message semantics ... 91
I.15 Video usability information ... 97

LIST OF TABLES

Table I.1 – Name association to prediction mode and partitioning type 33

Table I.2 – Specification of intra prediction mode and associated names 39

Table I.3 – Specification of divCoeff depending on sDenomDiv 67

Table I.4 – Association of ctxIdx and syntax elements for each initializationType in the initialization process 81

Table I.5 – Values of initValue for skip_intra_flag ctxIdx 81

Table I.6 – Values of initValue for no_dim_flag ctxIdx 81

Table I.7 – Values of initValue for depth_intra_mode_idx_flag ctxIdx 81

Table I.8 – Values of initValue for skip_intra_mode_idx ctxIdx 81

Table I.9 – Values of initValue for dbbp_flag ctxIdx 82

Table I.10 – Values of initValue for dc_only_flag ctxIdx 82

Table I.11 – Values of initValue for iv_res_pred_weight_idx ctxIdx 82

Table I.12 – Values of initValue for illu_comp_flag ctxIdx 82

Table I.13 – Values of initValue for depth_dc_present_flag ctxIdx 82

Table I.14 – Values of initValue for depth_dc_abs ctxIdx 82

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

Table I.15 – Syntax elements and associated binarizations 83

Table I.16 – Binarization for part_mode 84

Table I.17 – Assignment of ctxInc to syntax elements with context coded bins 86

Table I.18 – Specification of ctxInc using left and above syntax elements 86

Table I.19 – Persistence scope of SEI messages (informative) 91

Table I.20 – Interpretation of depth_type 92

Table I.21 – Locations of the top-left luma samples of constituent pictures packed in a picture with ViewIdx greater than 0 relative to the top-left luma sample of this picture 92

© ISO/IEC 2015 – All rights reserved

Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

Documents
Document Sharing Hub

Documents
Document Sharing Hub

Full document details: ISO_IEC 23008-2_2015_Amd 1_2015

Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding — Amendment 1

Introduction

Replace the text of 0.2 with the following:

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 Collaborative Team on Video Coding (JCT-VC) in 2010 and a Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) in 2012 for development of a new Recommendation | International Standard. This Recommendation | International Standard was developed in the JCT-VC and the JCT-3V.

In 0.3 add the following sentence to the end of the clause:

Support for 3D enables joint representation of video content and depth information with multiple camera views.

In 0.5 add the following paragraph to the end of the clause:

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 3 refers to the integrated text containing 3D extensions, additional supplement enhancement information, and corrections to various minor defects in the prior content of the specification.

In 0.8, replace "Annexes A through H" with "Annexes A through I".

In 0.8, add the following sentence after the sentence that starts with "Annex H":

Annex I contains support for 3D coding.

Sequence parameter set RBSP

In 7.3.2.2.1 and F.7.3.2.2.1 replace the row containing "if(sps_extension_present_flag)" and all following rows in the syntax table by the following:

if(sps_extension_present_flag) {	
sps_range_extension_flag	u(1)
sps_multilayer_extension_flag	u(1)
sps_3d_extension_flag	u(1)
sps_extension_5bits	u(5)
}	
if(sps_range_extension_flag)	
sps_range_extension( )	
if(sps_multilayer_extension_flag)	
sps_multilayer_extension( ) /* specified in Annex F */	
if(sps_3d_extension_flag)	
sps_3d_extension( ) /* specified in Annex I */	
if(sps_extension_5bits)	
while(more_rbsp_data( ))	
sps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

Documents
Document Sharing Hub

© ISO/IEC 2015 – All rights reserved

Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

In 7.4.3.2.1 replace the semantics of sps_extension_present_flag with the following semantics:

sps_extension_present_flag equal to 1 specifies that the syntax elements sps_range_extension_flag, sps_multilayer_extension_flag, sps_3d_extension_flag, and sps_extension_5bits are present in the SPS RBSP syntax structure. sps_extension_present_flag equal to 0 specifies that these syntax elements are not present.

In 7.4.3.2.1 add the following semantics after semantics of sps_multilayer_extension_flag:

sps_3d_extension_flag equal to 1 specifies that the sps_3d_extension( ) syntax structure (specified in Annex I) is present in the SPS RBSP syntax structure. sps_3d_extension_flag equal to 0 specifies that the sps_3d_extension( ) syntax structure is not present. When not present, the value of sps_3d_extension_flag is inferred to be equal to 0.

In 7.4.3.2.1 replace all occurrences of "sps_extension_6bits" with "sps_extension_5bits".

Picture parameter set RBSP

In 7.3.2.3.1 replace the row containing "if(pps_extension_present_flag)" and all following rows in the syntax table with the following:

if(pps_extension_present_flag) {	
pps_range_extension_flag	u(1)
pps_multilayer_extension_flag	u(1)
pps_3d_extension_flag	u(1)
pps_extension_5bits	u(5)
}	
if(pps_range_extension_flag)	
pps_range_extension( )	
if(pps_multilayer_extension_flag)	
pps_multilayer_extension( ) /* specified in Annex F */	
if(pps_3d_extension_flag)	
pps_3d_extension( ) /* specified in Annex I */	
if(pps_extension_5bits)	
while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

In 7.4.3.3.1 replace the semantics of pps_extension_present_flag with the following semantics:

pps_extension_present_flag equal to 1 specifies that the syntax elements pps_range_extension_flag, pps_multilayer_extension_flag, pps_3d_extension_flag, and pps_extension_5bits are present in the picture parameter set RBSP syntax structure. pps_extension_present_flag equal to 0 specifies that these syntax elements are not present.

In 7.4.3.3.1 add the following semantics after semantics of pps_multilayer_extension_flag:

pps_3d_extension_flag equal to 1 specifies that the pps_3d_extension( ) syntax structure (specified in Annex I) is present in the PPS RBSP syntax structure. pps_3d_extension_flag equal to 0 specifies that the pps_3d_extension( ) syntax structure is not present. When not present, the value of pps_3d_extension_flag is inferred to be equal to 0.

In 7.4.3.3.1 replace all occurrences of "pps_extension_6bits" with "pps_extension_5bits".

General SEI message syntax

In D.2.1 add the following rows to the syntax table after the row containing "multiview_view_position(payloadSize) / specified in Annex G */":*

else if(payloadType = = 181)	
alternative_depth_info(payloadSize) /* specified in Annex I */	

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

Common decoding process for a coded picture

In F.8.1.3 add the following paragraph before the paragraph starting with "After all slices of the current picture have been decoded,":

– Otherwise, general_profile_idc in the profile_tier_level( ) syntax structure VpsProfileTierLevel[profile_tier_level_idx[TargetOlsIdx][lIdx]] is equal to 8, the decoding process for the current picture takes as inputs the syntax elements and upper-case variables from clause I.7 and the decoding process of clause I.8.1.2 is invoked.

Video parameter set RBSP semantics

In F.7.4.3.1 replace the semantics of vps_extension2_flag with the following semantics:

vps_extension2_flag equal to 0 specifies that no vps_extension_data_flag syntax elements are present in the VPS RBSP syntax structure. vps_extension2_flag equal to 1 specifies vps_extension_data_flag syntax elements are present in the VPS RBSP syntax structure. Decoders conforming to a profile specified in Annexes A, G, or H shall ignore all data that follow the value 1 for vps_extension2_flag in a VPS NAL unit.

In F.7.4.3.1 add the following semantics:

vps_extension_data_flag may have any value. Its presence and value do not affect decoder conformance to profiles specified in Annexes A, G, or H. Decoders conforming to a profile specified in Annexes A, G, or H shall ignore all vps_extension_data_flag syntax elements.

In F.7.4.3.1.1 replace Table F.1 with the following table:

Table F.1 – Mapping of ScalabiltyId to scalability dimensions

scalability mask index	Scalability dimension	ScalabilityId mapping
0	Texture or depth	DepthLayerFlag
1	Multiview	ViewOrderIdx
2	Spatial/quality scalability	DependencyId
3	Auxiliary	AuxId
4-15	Reserved	

In F.7.4.3.1.1 remove Note 2.

In F.7.4.3.1.1 replace the paragraph starting with "The variable ScalabilityId[i][smIdx] specifying the identifier" and the following equation (F-2), with the following:

The variable ScalabilityId[i][smIdx] specifying the identifier of the smIdx-th scalability dimension type of the i-th layer, and the variables DepthLayerFlag[lId], ViewOrderIdx[lId], DependencyId[lId], and AuxId[lId] specifying the depth flag, view order index, the spatial/quality scalability identifier, and the auxiliary identifier, respectively, of the layer with nuh_layer_id equal to lId are derived as follows:

```
NumViews = 1
for( i = 0; i  <=  MaxLayersMinus1; i++ ) {
    lId = layer_id_in_nuh[ i ]
    for( smIdx= 0, j = 0; smIdx < 16; smIdx++ ) {
        if( scalability_mask_flag[ smIdx ] )
            ScalabilityId[ i ][ smIdx ] = dimension_id[ i ][ j++ ]
        else
            ScalabilityId[ i ][ smIdx ] = 0
    }
    DepthLayerFlag[ lId ] = ScalabilityId[ i ][ 0 ]
    ViewOrderIdx[ lId ] = ScalabilityId[ i ][ 1 ]
    DependencyId[ lId ] = ScalabilityId[ i ][ 2 ]                    (F-2)
    AuxId[ lId ] = ScalabilityId[ i ][ 3 ]
    if( i > 0 ) {
        newViewFlag = 1
```


© ISO/IEC 2015 – All rights reserved

Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

```
                for( j = 0; j < i; j++ )
                    if( ViewOrderIdx[ lId ]  = =  ViewOrderIdx[ layer_id_in_nuh[ j ] ] )
                        newViewFlag = 0
                NumViews  +=  newViewFlag
            }
        }
```

In F.7.4.3.1.1 replace the semantics of direct_dep_type_len_minus2 with the following:

direct_dep_type_len_minus2 plus 2 specifies the number of bits of the direct_dependency_type[i][j] and the direct_dependency_all_layers_type syntax elements. In bitstreams conforming to this version of this Specification the value of direct_dep_type_len_minus2 shall be equal 0 or 1. Although the value of direct_dep_type_len_minus2 shall be equal to 0 or 1 in this version of this Specification, decoders shall allow other values of direct_dep_type_len_minus2 in the range of 0 to 30, inclusive, to appear in the syntax.

In F.7.4.3.1.1 replace the semantics of direct_dependency_all_layers_type with the following:

direct_dependency_all_layers_type, when present, specifies the inferred value of direct_dependency_type[i][j] for all combinations of i-th and j-th layers. The length of the direct_dependency_all_layers_type syntax element is direct_dep_type_len_minus2 + 2 bits. Although the value of direct_dependency_all_layers_type is required to be in the range of 0 to 6, inclusive, in this version of this Specification, decoders shall allow values of direct_dependency_all_layers_type in the range of 0 to $2^{32} - 2$, inclusive, to appear in the syntax.

In F.7.4.3.1.1 replace the semantics of direct_dependency_type with the following:

direct_dependency_type[i][j] indicates the type of dependency between the layer with nuh_layer_id equal layer_id_in_nuh[i] and the layer with nuh_layer_id equal to layer_id_in_nuh[j]. direct_dependency_type[i][j] equal to 0 specifies that the layer with nuh_layer_id equal to layer_id_in_nuh[j] may be used for inter-layer sample prediction but is not used for inter-layer motion prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. direct_dependency_type[i][j] equal to 1 specifies that the layer with nuh_layer_id equal to layer_id_in_nuh[j] may be used for inter-layer motion prediction but is not used for inter-layer sample prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. direct_dependency_type[i][j] equal to 2 specifies that the layer with nuh_layer_id equal to layer_id_in_nuh[j] may be used for both inter-layer motion prediction and inter-layer sample prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. The length of the direct_dependency_type[i][j] syntax element is direct_dep_type_len_minus2 + 2 bits. Although the value of direct_dependency_type[i][j] shall be in the range of 0 to 2, inclusive, when the layer with nuh_layer_id equal to layer_id_in_nuh[i] conforms to a profile specified in Annexes A, G, or H, and in the range of 0 to 6, inclusive, when the layer with nuh_layer_id equal to layer_id_in_nuh[i] conforms to a profile specified in Annex I, decoders shall allow values of direct_dependency_type[i][j] in the range of 0 to $2^{32} - 2$, inclusive, to appear in the syntax.

Profiles, tiers, and levels

In F.11.2 add the following row to the end of Table F.3:

3D Main	3D Main, Multiview Main, Main, Main Still Picture

In G.11.1.1 and H.11.1.1 remove " and sps_extension_6bits equal to 0 only".

In G.11.1.1 and H.11.1.1 remove " and pps_extension_6bits equal to 0 only".

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

Add a new Annex I as follows:

Annex I

3D high efficiency video coding

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

I.1 Scope

This annex specifies syntax, semantics, and decoding processes for 3D high efficiency video coding that use the syntax, semantics, and decoding processes specified in clauses 2-10 and Annexes A-G. This annex also specifies profiles, tiers, and levels for 3D high efficiency video coding.

I.2 Normative references

The list of normative references in clause G.2 apply.

I.3 Definitions

For the purpose of this annex, the following definitions apply in addition to the definitions in clause G.3. These definitions are either not present in clause G.3 or replace definitions in clause G.3.

I.3.1 **depth intra contour prediction**: A *prediction* of a *partition pattern* for a *prediction block* in a *picture* of a *depth layer* derived from samples of a *picture* included in the same *access unit* and in the *texture layer* of the same *view*.

I.3.2 **depth layer**: A *layer* with a nuh_layer_id value equal to i, such that DepthLayerFlag[i] is equal to 1 and DependencyId[i] and AuxId[i] are equal to 0.

I.3.3 **depth look-up table**: A list containing *depth values*.

I.3.4 **depth value**: A sample value of a *decoded picture* of a *depth layer*.

I.3.5 **disparity vector**: A *motion vector* used for inter-view prediction.

I.3.6 **inter-component prediction**: An *inter-layer prediction* where the *reference pictures* are associated with a DepthFlag value different from the DepthFlag value of the current *picture*.

I.3.7 **inter-view prediction**: An *inter-layer prediction* where the *reference pictures* are associated with *reference view order index* values different from the ViewIdx value of the current *picture*.

I.3.8 **intra prediction**: A *prediction* derived from only data elements (e.g., sample values) of the same decoded *slice* and additionally *may* be using *depth intra contour prediction*.

I.3.9 **partition pattern**: An MxM (M-column by M-row) array of *flags* defining two *sub-block partitions* of an MxM *prediction block*.

I.3.10 **prediction block**: A rectangular MxN *block* of samples on which either the same *prediction* or *partitioning* in *sub-block partitions* is applied.

I.3.11 **reference view order index**: A ViewIdx value associated with a *reference picture* used for *inter-view prediction*.

I.3.12 **sub-block partition**: A subset of samples of a *prediction block* on which the same *prediction* is applied.

I.3.13 **texture layer**: A *layer* with a nuh_layer_id value equal to i, such that DepthLayerFlag[i], DependencyId[i], and AuxId[i] are equal to 0.

I.4 Abbreviations

The specifications in clause G.4 apply.

I.5 Conventions

The specifications in clause G.5 apply.

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

I.6 Bitstream and picture formats, partitionings, scanning processes, and neighbouring relationships

I.6.1 Bitstream formats

The specifications in clause 6.1 apply.

I.6.2 Source, decoded, and output picture formats

The specifications in clause 6.2 apply.

I.6.3 Partitioning of pictures, slices, slice segments, tiles, coding tree units, and coding tree blocks

The specifications in clause 6.3 and its subclauses apply.

I.6.4 Availability processes

The specifications in clause 6.4 apply.

I.6.5 Scanning processes

The specifications in clause 6.5 and its subclauses apply.

I.6.6 Derivation process for a wedgelet partition pattern table

NOTE – Tables and values resulting from this process are independent of any information contained in the bitstream.

The list WedgePatternTable[log2BlkSize] of partition patterns of size (1 << log2BlkSize)x(1 << log2BlkSize) and the variable NumWedgePattern[log2BlkSize] specifying the number of partition patterns in list WedgePatternTable[log2BlkSize] are derived as follows:

- For log2BlkSize in the range of 2 to 4, inclusive, the following applies:
  - NumWedgePattern[log2BlkSize] is set equal to 0.
  - The variable resShift is set equal to (log2BlkSize == 4) ? 0 : 1.
  - The variable wBlkSize is set equal to (1 << (log2BlkSize + resShift)).
  - For wedgeOri in the range of 0 to 5, inclusive, the following applies:
    - The variable posEnd is set equal to NumWedgePattern[log2BlkSize].
    - If wedgeOri is equal to 0 or 4, the following applies:
      - The variables sizeScaleS and sizeScaleE are derived as follows:

$$\text{sizeScaleS} = (\text{log2BlkSize} > 3) \ ? \ 2 : 1 \qquad \text{(I–1)}$$

$$\text{sizeScaleE} = (\text{wedgeOri} < 4 \ \&\& \ \text{log2BlkSize} > 3) \ ? : 2 : 1 \qquad \text{(I–2)}$$

      - For m in the range of 0 to (wBlkSize / sizeScaleS − 1), inclusive, the following applies:
        - For n in the range of 0 to (wBlkSize / sizeScaleE − 1), inclusive, the following applies:
          - The wedgelet partition pattern generation process as specified in clause I.6.6.1 is invoked with patternSize equal to (1 << log2BlkSize), the variable resShift, the variable wedgeOri, the variable (xS, yS) equal to (m * sizeScaleS , 0), the variable (xE, yE) equal to (wedgeOri == 0) ? (0, n * sizeScaleE) : (n * sizeScaleE, wBlkSize − 1) as inputs, and the output is the partition pattern curWedgePattern.
          - The wedgelet partition pattern table insertion process as specified in clause I.6.6.2 is invoked with the variable log2BlkSize and the partition pattern curWedgePattern as inputs.
    - Otherwise (wedgeOri is equal to 1, 2, 3, or 5), the following applies:
      - For curPos in the range of posStart to posEnd − 1, inclusive, the following applies:
        - The partition pattern curWedgePattern[x][y] is derived as follows:

```
for( y = 0; y < ( 1 << log2BlkSize ); y++ )
    for( x = 0; x < ( 1 << log2BlkSize ); x++ )                    (I–3)
        curWedgePattern[ x ][ y ] = 1 −
            WedgePatternTable[ log2BlkSize ][ curPos ][ y ][ ( 1 << log2BlkSize ) − 1 − x ]
```

    - The variable posStart is set equal to posEnd.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

– NumWedgePattern[5] is set equal to NumWedgePattern[4].

– For k = 0..NumWedgePattern[5] − 1, the following applies:

  – For x, y = 0..(1 << 5) − 1, the following applies:

$$\text{WedgePatternTable[5][k][x][y] = WedgePatternTable[4][k][x >> 1][y >> 1]} \quad \text{(I−4)}$$

I.6.6.1 Wedgelet partition pattern generation process

Inputs to this process are:

– a variable patternSize specifying the partition pattern size,

– a variable resShift specifying the precision of the partition pattern start and end locations relative to patternSize,

– a variable wedgeOri specifying the orientation of the partition pattern,

– a location (xS, yS) specifying the boundary start of a sub-block partition,

– a location (xE, yE) specifying the boundary end of a sub-block partition.

Output of this process is the partition pattern wedgePattern[x][y] of size (patternSize)x(patternSize).

The values of the partition pattern wedgePattern[x][y] are derived as specified by the following ordered steps:

1. For x, y = 0..patternSize − 1, wedgePattern[x][y] is set equal to 0.

2. The samples of the partition pattern wedgePattern that form a line between (xS, yS) and (xE, yE) are set equal to 1 as follows:

```
( x0, y0 ) = ( xS, yS )
( x1, y1 ) = ( xE, yE )
if( abs( yE − yS ) > abs( xE − xS ) ) {
    ( x0, y0 ) = Swap( x0, y0 )
    ( x1, y1 ) = Swap( x1, y1 )
}
if( x0 > x1 ) {
    ( x0, x1 ) = Swap( x0, x1 )
    ( y0, y1 ) = Swap( y0, y1 )
}
sumErr = 0                                                    (I−5)
posY = y0
for( posX = x0; posX <= x1; posX++ ) {
    if( abs( yE − yS ) > abs( xE − xS ) )
        wedgePattern[ posY >> resShift ][ posX >> resShift ] = 1
    else
        wedgePattern[ posX >> resShift ][ posY >> resShift ] = 1
    sumErr += ( abs( y1 − y0 ) << 1 )
    if( sumErr >= ( x1 − x0 ) ) {
        posY += ( y0 < y1 ) ? 1 : −1
        sumErr −= ( x1 − x0 ) << 1
    }
}
```

3. The samples of wedgePattern are modified as follows:

```
for( y = 0; y <= ( yE >> resShift ); y++ )
    for( x = 0; ( x <= patternSize − 1 ) && ( wedgePattern[ x ][ y ] == 0 ); x++ )   (I−6)
        wedgePattern[ x ][ y ] = 1
```

I.6.6.2 Wedgelet partition pattern table insertion process

Inputs to this process are:

– a variable log2BlkSize specifying the partition pattern size,

– a partition pattern wedgePattern[x][y], with x, y = 0..(1 << log2BlkSize) − 1.

The variable validPatternFlag is set equal to 0 and the following applies:

1. For x, y = 0..(1 << log2BlkSize) − 1, the following applies:

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

  – When wedgePattern[x][y] is not equal to wedgePattern[0][0], validPatternFlag is set equal to 1.

2. For k = 0..NumWedgePattern[log2BlkSize] − 1, the following applies:
  – The variable patIdenticalFlag is set equal to 1.
  – For x, y = 0..(1 << log2BlkSize) − 1, the following applies:
    – When wedgePattern[x][y] is not equal to WedgePatternTable[log2BlkSize][k][x][y], patIdenticalFlag is set equal to 0.
  – When patIdenticalFlag is equal to 1, validPatternFlag is set equal to 0.
3. For k = 0..NumWedgePattern[log2BlkSize] − 1, the following applies:
  – The variable patInvIdenticalFlag is set equal to 1.
  – For x, y = 0..(1 << log2BlkSize) − 1, the following applies:
    – When wedgePattern[x][y] is equal to WedgePatternTable[log2BlkSize][k][x][y], patInvIdenticalFlag is set equal to 0.
  – When patInvIdenticalFlag is equal to 1, validPatternFlag is set equal to 0.

When validPatternFlag is equal to 1, the following applies:

– The pattern WedgePatternTable[log2BlkSize][NumWedgePattern[log2BlkSize]] is set equal to wedgePattern.
– The value of NumWedgePattern[log2BlkSize] is incremented by one.

I.7 Syntax and semantics

I.7.1 Method of specifying syntax in tabular form

The specifications in clause F.7.1 apply.

I.7.2 Specification of syntax functions, categories, and descriptors

The specifications in clause F.7.2 apply.

I.7.3 Syntax in tabular form

I.7.3.1 NAL unit syntax

The specifications in clause F.7.3.1 and all its subclauses apply.

I.7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax

I.7.3.2.1 Video parameter set RBSP

video_parameter_set_rbsp( ) {	**Descriptor**
vps_video_parameter_set_id	u(4)
vps_base_layer_internal_flag	u(1)
vps_base_layer_available_flag	u(1)
vps_max_layers_minus1	u(6)
vps_max_sub_layers_minus1	u(3)
vps_temporal_id_nesting_flag	u(1)
vps_reserved_0xffff_16bits	u(16)
profile_tier_level(1, vps_max_sub_layers_minus1)	
vps_sub_layer_ordering_info_present_flag	u(1)
for(i = (vps_sub_layer_ordering_info_present_flag ? 0 : vps_max_sub_layers_minus1); i <= vps_max_sub_layers_minus1; i++) {	
vps_max_dec_pic_buffering_minus1[i]	ue(v)
vps_max_num_reorder_pics[i]	ue(v)
vps_max_latency_increase_plus1[i]	ue(v)
}	

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

vps_max_layer_id	u(6)
vps_num_layer_sets_minus1	ue(v)
for(i = 1; i <= vps_num_layer_sets_minus1; i++)	
for(j = 0; j <= vps_max_layer_id; j++)	
layer_id_included_flag[i][j]	u(1)
vps_timing_info_present_flag	u(1)
if(vps_timing_info_present_flag) {	
vps_num_units_in_tick	u(32)
vps_time_scale	u(32)
vps_poc_proportional_to_timing_flag	u(1)
if(vps_poc_proportional_to_timing_flag)	
vps_num_ticks_poc_diff_one_minus1	ue(v)
vps_num_hrd_parameters	ue(v)
for(i = 0; i < vps_num_hrd_parameters; i++) {	
hrd_layer_set_idx[i]	ue(v)
if(i > 0)	
cprms_present_flag[i]	u(1)
hrd_parameters(cprms_present_flag[i], vps_max_sub_layers_minus1)	
}	
}	
vps_extension_flag	u(1)
if(vps_extension_flag) {	
while(!byte_aligned( ))	
vps_extension_alignment_bit_equal_to_one	u(1)
vps_extension( )	
vps_extension2_flag	u(1)
if(vps_extension2_flag) {	
vps_3d_extension_flag	u(1)
if(vps_3d_extension_flag) {	
while(!byte_aligned( ))	
vps_3d_extension_alignment_bit_equal_to_one	u(1)
vps_3d_extension( )	
}	
vps_extension3_flag	u(1)
if(vps_extension3_flag)	
while(more_rbsp_data( ))	
vps_extension_data_flag	u(1)
}	
}	
rbsp_trailing_bits( )	
}	

I.7.3.2.1.1 Video parameter set extension syntax

The specifications in clause F.7.3.2.1.1 apply.

I.7.3.2.1.2 Representation format syntax

The specifications in clause F.7.3.2.1.2 apply.

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

I.7.3.2.1.3 DPB size syntax

The specifications in clause F.7.3.2.1.3 apply.

I.7.3.2.1.4 VPS VUI syntax

The specifications in clause F.7.3.2.1.4 apply.

I.7.3.2.1.5 Video signal info syntax

The specifications in clause F.7.3.2.1.5 apply.

I.7.3.2.1.6 VPS VUI bitstream partition HRD parameters syntax

The specifications in clause F.7.3.2.1.6 apply.

I.7.3.2.1.7 Video parameter set 3D extension syntax

vps_3d_extension( ) {	**Descriptor**
cp_precision	ue(v)
for(n = 1; n < NumViews; n++) {	
i = ViewOIdxList[n]	
num_cp[i]	u(6)
if(num_cp[i] > 0) {	
cp_in_slice_segment_header_flag[i]	u(1)
for(m = 0; m < num_cp[i]; m++) {	
cp_ref_voi[i][m]	ue(v)
if(!cp_in_slice_segment_header_flag[i]) {	
j = cp_ref_voi[i][m]	
vps_cp_scale[i][j]	se(v)
vps_cp_off[i][j]	se(v)
vps_cp_inv_scale_plus_scale[i][j]	se(v)
vps_cp_inv_off_plus_off[i][j]	se(v)
}	
}	
}	
}	
}	

I.7.3.2.2 Sequence parameter set RBSP syntax

I.7.3.2.2.1 General sequence parameter set RBSP syntax

The specifications in clause F.7.3.2.2.1 apply.

I.7.3.2.2.2 Sequence parameter set range extensions syntax

The specifications in clause F.7.3.2.2.2 apply.

I.7.3.2.2.3 Sequence parameter set multilayer extension syntax

The specifications in clause F.7.3.2.2.3 apply.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

I.7.3.2.2.4 Sequence parameter set 3D extension syntax

sps_3d_extension( ) {	Descriptor
for(d = 0; d <= 1; d++) {	
iv_di_mc_enabled_flag[d]	u(1)
iv_mv_scal_enabled_flag[d]	u(1)
if(d == 0) {	
log2_ivmc_sub_pb_size_minus3[d]	ue(v)
iv_res_pred_enabled_flag[d]	u(1)
depth_ref_enabled_flag[d]	u(1)
vsp_mc_enabled_flag[d]	u(1)
dbbp_enabled_flag[d]	u(1)
} else {	
tex_mc_enabled_flag[d]	u(1)
log2_texmc_sub_pb_size_minus3[d]	ue(v)
intra_contour_enabled_flag[d]	u(1)
intra_dc_only_wedge_enabled_flag[d]	u(1)
cqt_cu_part_pred_enabled_flag[d]	u(1)
inter_dc_only_enabled_flag[d]	u(1)
skip_intra_enabled_flag[d]	u(1)
}	
}	
}	

I.7.3.2.3 Picture parameter set RBSP syntax

I.7.3.2.3.1 General picture parameter set RBSP syntax

The specifications in clause F.7.3.2.3.1 apply.

I.7.3.2.3.2 Picture parameter set range extensions syntax

The specifications in clause F.7.3.2.3.2 apply.

I.7.3.2.3.3 Picture parameter set multilayer extension syntax

The specifications in clause F.7.3.2.3.3 apply.

I.7.3.2.3.4 General colour mapping table syntax

The specifications in clause F.7.3.2.3.4 apply.

I.7.3.2.3.5 Colour mapping octants syntax

The specifications in clause F.7.3.2.3.5 apply.

I.7.3.2.3.6 Picture parameter set 3D extension syntax

pps_3d_extension( ) {	Descriptor
dlts_present_flag	u(1)
if(dlts_present_flag) {	
pps_depth_layers_minus1	u(6)
pps_bit_depth_for_depth_layers_minus8	u(4)
for(i = 0; i <= pps_depth_layers_minus1; i++) {	
dlt_flag[i]	u(1)

AVA Documents
Document Sharing Hub

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

if(dlt_flag[i]) {	
dlt_pred_flag[i]	u(1)
if(!dlt_pred_flag[i])	
dlt_val_flags_present_flag[i]	u(1)
if(dlt_val_flags_present_flag[i])	
for(j = 0; j <= depthMaxValue; j++)	
dlt_value_flag[i][j]	u(1)
else	
delta_dlt(i)	
}	
}	
}	
}	

I.7.3.2.3.7 Delta depth look-up table syntax

delta_dlt(i) {	**Descriptor**
num_val_delta_dlt	u(v)
if(num_val_delta_dlt > 0) {	
if(num_val_delta_dlt > 1)	
max_diff	u(v)
if(num_val_delta_dlt > 2 && max_diff > 0)	
min_diff_minus1	u(v)
delta_dlt_val0	u(v)
if(max_diff > (min_diff_minus1 + 1))	
for(k = 1; k < num_val_delta_dlt; k++)	
delta_val_diff_minus_min[k]	u(v)
}	
}	

I.7.3.2.4 Supplemental enhancement information RBSP syntax

The specifications in clause F.7.3.2.4 apply.

I.7.3.2.5 Access unit delimiter RBSP syntax

The specifications in clause F.7.3.2.5 apply.

I.7.3.2.6 End of sequence RBSP syntax

The specifications in clause F.7.3.2.6 apply.

I.7.3.2.7 End of bitstream RBSP syntax

The specifications in clause F.7.3.2.7 apply.

I.7.3.2.8 Filler data RBSP syntax

The specifications in clause F.7.3.2.8 apply.

I.7.3.2.9 Slice segment layer RBSP syntax

The specifications in clause F.7.3.2.9 apply.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

I.7.3.2.10 RBSP slice segment trailing bits syntax

The specifications in clause F.7.3.2.10 apply.

I.7.3.2.11 RBSP trailing bits syntax

The specifications in clause F.7.3.2.11 apply.

I.7.3.2.12 Byte alignment syntax

The specifications in clause F.7.3.2.12 apply.

I.7.3.3 Profile, tier and level syntax

The specifications in clause F.7.3.3 apply.

I.7.3.4 Scaling list data syntax

The specifications in clause F.7.3.4 apply.

I.7.3.5 Supplemental enhancement information message syntax

The specifications in clause F.7.3.5 apply.

I.7.3.6 Slice segment header syntax

I.7.3.6.1 General slice segment header syntax

slice_segment_header( ) {	**Descriptor**
first_slice_segment_in_pic_flag	u(1)
if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
no_output_of_prior_pics_flag	u(1)
slice_pic_parameter_set_id	ue(v)
if(!first_slice_segment_in_pic_flag) {	
if(dependent_slice_segments_enabled_flag)	
dependent_slice_segment_flag	u(1)
slice_segment_address	u(v)
}	
if(!dependent_slice_segment_flag) {	
i = 0	
if(num_extra_slice_header_bits > i) {	
i++	
discardable_flag	u(1)
}	
if(num_extra_slice_header_bits > i) {	
i++	
cross_layer_bla_flag	u(1)
}	
for(; i < num_extra_slice_header_bits; i++)	
slice_reserved_flag[i]	u(1)
slice_type	ue(v)
if(output_flag_present_flag)	
pic_output_flag	u(1)
if(separate_colour_plane_flag = = 1)	
colour_plane_id	u(2)
if((nuh_layer_id > 0 && !poc_lsb_not_present_flag[LayerIdxInVps[nuh_layer_id]]) || (nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP))	

© ISO/IEC 2015 – All rights reserved

Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

slice_pic_order_cnt_lsb	u(v)
if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) {	
short_term_ref_pic_set_sps_flag	u(1)
if(!short_term_ref_pic_set_sps_flag)	
st_ref_pic_set(num_short_term_ref_pic_sets)	
else if(num_short_term_ref_pic_sets > 1)	
short_term_ref_pic_set_idx	u(v)
if(long_term_ref_pics_present_flag) {	
if(num_long_term_ref_pics_sps > 0)	
num_long_term_sps	ue(v)
num_long_term_pics	ue(v)
for(i = 0; i < num_long_term_sps + num_long_term_pics; i++) {	
if(i < num_long_term_sps) {	
if(num_long_term_ref_pics_sps > 1)	
lt_idx_sps[i]	u(v)
} else {	
poc_lsb_lt[i]	u(v)
used_by_curr_pic_lt_flag[i]	u(1)
}	
delta_poc_msb_present_flag[i]	u(1)
if(delta_poc_msb_present_flag[i])	
delta_poc_msb_cycle_lt[i]	ue(v)
}	
}	
if(sps_temporal_mvp_enabled_flag)	
slice_temporal_mvp_enabled_flag	u(1)
}	
if(nuh_layer_id > 0 && !default_ref_layers_active_flag && NumRefListLayers[nuh_layer_id] > 0) {	
inter_layer_pred_enabled_flag	u(1)
if(inter_layer_pred_enabled_flag && NumRefListLayers[nuh_layer_id] > 1) {	
if(!max_one_active_ref_layer_flag)	
num_inter_layer_ref_pics_minus1	u(v)
if(NumActiveRefLayerPics != NumRefListLayers[nuh_layer_id])	
for(i = 0; i < NumActiveRefLayerPics; i++)	
inter_layer_pred_layer_idc[i]	u(v)
}	
}	
if(inCmpPredAvailFlag)	
in_comp_pred_flag	u(1)
if(sample_adaptive_offset_enabled_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(slice_type == P || slice_type == B) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag) {	
num_ref_idx_l0_active_minus1	ue(v)

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

if(slice_type = = B)	
num_ref_idx_l1_active_minus1	ue(v)
}	
if(lists_modification_present_flag && NumPicTotalCurr > 1)	
ref_pic_lists_modification( )	
if(slice_type = = B)	
mvd_l1_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(slice_type = = B)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) || (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
collocated_ref_idx	ue(v)
}	
if((weighted_pred_flag && slice_type = = P) || (weighted_bipred_flag && slice_type = = B))	
pred_weight_table( )	
else if(!DepthFlag && NumRefListLayers[nuh_layer_id] > 0) {	
slice_ic_enabled_flag	u(1)
if(slice_ic_enabled_flag)	
slice_ic_disabled_merge_zero_idx_flag	u(1)
}	
five_minus_max_num_merge_cand	ue(v)
}	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
}	
if(chroma_qp_offset_list_enabled_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(pps_loop_filter_across_slices_enabled_flag && (slice_sao_luma_flag || slice_sao_chroma_flag || !slice_deblocking_filter_disabled_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
if(cp_in_slice_segment_header_flag[ViewIdx])	
for(m = 0; m < num_cp[ViewIdx]; m++) {	
j = cp_ref_voi[ViewIdx][m]	

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

cp_scale[j]	se(v)
cp_off[j]	se(v)
cp_inv_scale_plus_scale[j]	se(v)
cp_inv_off_plus_off[j]	se(v)
}	
}	
if(tiles_enabled_flag || entropy_coding_sync_enabled_flag) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset_minus1[i]	u(v)
}	
}	
if(slice_segment_header_extension_present_flag) {	
slice_segment_header_extension_length	ue(v)
if(poc_reset_info_present_flag)	
poc_reset_idc	u(2)
if(poc_reset_idc != 0)	
poc_reset_period_id	u(6)
if(poc_reset_idc == 3) {	
full_poc_reset_flag	u(1)
poc_lsb_val	u(v)
}	
if(!PocMsbValRequiredFlag && vps_poc_lsb_aligned_flag)	
poc_msb_cycle_val_present_flag	u(1)
if(poc_msb_cycle_val_present_flag)	
poc_msb_cycle_val	ue(v)
while(more_data_in_slice_segment_header_extension( ))	
slice_segment_header_extension_data_bit	u(1)
}	
byte_alignment( )	
}	

I.7.3.6.2 Reference picture list modification syntax

The specifications in clause F.7.3.6.2 apply.

I.7.3.6.3 Weighted prediction parameters syntax

The specifications in clause F.7.3.6.3 apply.

I.7.3.7 Short-term reference picture set syntax

The specifications in clause F.7.3.7 apply.

I.7.3.8 Slice segment data syntax

I.7.3.8.1 General slice segment data syntax

The specifications in clause F.7.3.8.1 apply.

I.7.3.8.2 Coding tree unit syntax

The specifications in clause F.7.3.8.2 apply.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

I.7.3.8.3 Sample adaptive offset syntax

The specifications in clause F.7.3.8.3 apply.

I.7.3.8.4 Coding quadtree syntax

coding_quadtree(x0, y0, log2CbSize, cqtDepth) {	**Descriptor**
if(x0 + (1 << log2CbSize) <= pic_width_in_luma_samples && y0 + (1 << log2CbSize) <= pic_height_in_luma_samples && log2CbSize > MinCbLog2SizeY && !predSplitCuFlag)	
split_cu_flag[x0][y0]	ae(v)
if(cu_qp_delta_enabled_flag && log2CbSize >= Log2MinCuQpDeltaSize) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
}	
if(cu_chroma_qp_offset_enabled_flag &&	
log2CbSize >= Log2MinCuChromaQpOffsetSize)	
IsCuChromaQpOffsetCoded = 0	
if(split_cu_flag[x0][y0]) {	
x1 = x0 + (1 << (log2CbSize − 1))	
y1 = y0 + (1 << (log2CbSize − 1))	
coding_quadtree(x0, y0, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples)	
coding_quadtree(x1, y0, log2CbSize − 1, cqtDepth + 1)	
if(y1 < pic_height_in_luma_samples)	
coding_quadtree(x0, y1, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_quadtree(x1, y1, log2CbSize − 1, cqtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbSize)	
}	

I.7.3.8.5 Coding unit syntax

coding_unit(x0, y0, log2CbSize) {	**Descriptor**
if(transquant_bypass_enabled_flag)	
cu_transquant_bypass_flag	ae(v)
if(slice_type != I)	
cu_skip_flag[x0][y0]	ae(v)
nCbS = (1 << log2CbSize)	
if(cu_skip_flag[x0][y0])	
prediction_unit(x0, y0, nCbS, nCbS)	
else if(SkipIntraEnabledFlag)	
skip_intra_flag[x0][y0]	ae(v)
if(!cu_skip_flag[x0][y0] && !skip_intra_flag[x0][y0]) {	
if(slice_type != I)	
pred_mode_flag	ae(v)
if((CuPredMode[x0][y0] != MODE_INTRA || log2CbSize == MinCbLog2SizeY) && !predPartModeFlag)	
part_mode	ae(v)

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
AVA Documents Document Sharing Hub

if(CuPredMode[x0][y0] = = MODE_INTRA) {	
if(PartMode = = PART_2Nx2N && pcm_enabled_flag && log2CbSize >= Log2MinIpcmCbSizeY && log2CbSize <= Log2MaxIpcmCbSizeY)	
pcm_flag[x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned( ))	
pcm_alignment_zero_bit	f(1)
pcm_sample(x0, y0, log2CbSize)	
} else {	
pbOffset = (PartMode = = PART_NxN) ? (nCbS / 2) : nCbS	
log2PbSize = log2CbSize − ((PartMode = = PART_NxN) ? 1 : 0)	
for(j = 0; j < nCbS; j = j + pbOffset)	
for(i = 0; i < nCbS; i = i + pbOffset) {	
if(IntraDcOnlyWedgeEnabledFlag || IntraContourEnabledFlag)	
intra_mode_ext(x0 + i , y0 + j , log2PbSize)	
if(no_dim_flag[x0 + i][y0 + j])	
prev_intra_luma_pred_flag[x0 + i][y0 + j]	ae(v)
}	
for(j = 0; j < nCbS; j = j + pbOffset)	
for(i = 0; i < nCbS; i = i + pbOffset)	
if(no_dim_flag[x0 + i][y0 + j]) {	
if(prev_intra_luma_pred_flag[x0 + i][y0 + j])	
mpm_idx[x0 + i][y0 + j]	ae(v)
else	
rem_intra_luma_pred_mode[x0 + i][y0 + j]	ae(v)
}	
if(ChromaArrayType = = 3)	
for(j = 0; j < nCbS; j = j + pbOffset)	
for(i = 0; i < nCbS; i = i + pbOffset)	
intra_chroma_pred_mode[x0 + 1][y0 + j]	ae(v)
else if(ChromaArrayType != 0)	
intra_chroma_pred_mode[x0][y0]	ae(v)
}	
} else {	
if(PartMode = = PART_2Nx2N)	
prediction_unit(x0, y0, nCbS, nCbS)	
else if(PartMode = = PART_2NxN) {	
prediction_unit(x0, y0, nCbS, nCbS / 2)	
prediction_unit(x0, y0 + (nCbS / 2), nCbS, nCbS / 2)	
} else if(PartMode = = PART_Nx2N) {	
prediction_unit(x0, y0, nCbS / 2, nCbS)	
prediction_unit(x0 + (nCbS / 2), y0, nCbS / 2, nCbS)	
} else if(PartMode = = PART_2NxnU) {	
prediction_unit(x0, y0, nCbS, nCbS / 4)	
prediction_unit(x0, y0 + (nCbS / 4), nCbS, nCbS * 3 / 4)	
} else if(PartMode = = PART_2NxnD) {	
prediction_unit(x0, y0, nCbS, nCbS * 3 / 4)	
prediction_unit(x0, y0 + (nCbS * 3 / 4), nCbS, nCbS / 4)	

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

} else if(PartMode = = PART_nLx2N) {	
prediction_unit(x0, y0, nCbS / 4, nCbS)	
prediction_unit(x0 + (nCbS / 4), y0, nCbS * 3 / 4, nCbS)	
} else if(PartMode = = PART_nRx2N) {	
prediction_unit(x0, y0, nCbS * 3 / 4, nCbS)	
prediction_unit(x0 + (nCbS * 3 / 4), y0, nCbS / 4, nCbS)	
} else { /* PART_NxN */	
prediction_unit(x0, y0, nCbS / 2, nCbS / 2)	
prediction_unit(x0 + (nCbS / 2), y0, nCbS / 2, nCbS / 2)	
prediction_unit(x0, y0 + (nCbS / 2), nCbS / 2, nCbS / 2)	
prediction_unit(x0 + (nCbS / 2), y0 + (nCbS / 2), nCbS / 2, nCbS / 2)	
}	
}	
}	
cu_extension(x0, y0, log2CbSize)	
if(DcOnlyFlag[x0][y0] || (!skip_intra_flag[x0][y0] && CuPredMode[x0][y0] = = MODE_INTRA))	
depth_dcs(x0, y0, log2CbSize)	
if(!cu_skip_flag[x0][y0] && !skip_intra_flag[x0][y0] && !dc_only_flag[x0][y0] && !pcm_flag[x0][y0]) {	
if(CuPredMode[x0][y0] != MODE_INTRA && !(PartMode = = PART_2Nx2N && merge_flag[x0][y0]))	
rqt_root_cbf	ae(v)
if(rqt_root_cbf) {	
MaxTrafoDepth = (CuPredMode[x0][y0] = = MODE_INTRA ? (max_transform_hierarchy_depth_intra + IntraSplitFlag) : max_transform_hierarchy_depth_inter)	
transform_tree(x0, y0, x0, y0, log2CbSize, 0, 0)	
}	
}	
}	

I.7.3.8.5.1 Intra mode extension syntax

intra_mode_ext(x0, y0, log2PbSize) {	**Descriptor**
if(log2PbSize < 6)	
no_dim_flag[x0][y0]	ae(v)
if(!no_dim_flag[x0][y0] && IntraDcOnlyWedgeEnabledFlag && IntraContourEnabledFlag)	
depth_intra_mode_idx_flag[x0][y0]	ae(v)
if(!no_dim_flag[x0][y0] && !depth_intra_mode_idx_flag[x0][y0])	
wedge_full_tab_idx[x0][y0]	ae(v)
}	

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

I.7.3.8.5.2 Coding unit extension syntax

cu_extension(x0, y0, log2CbSize) {	**Descriptor**
if(skip_intra_flag[x0][y0])	
skip_intra_mode_idx[x0][y0]	ae(v)
else {	
if(!cu_skip_flag[x0][y0]) {	
if(DbbpEnabledFlag && DispAvailFlag && log2CbSize > 3 && (PartMode == PART_2NxN || PartMode == PART_Nx2N))	
dbbp_flag[x0][y0]	ae(v)
if((CuPredMode[x0][y0] == MODE_INTRA ? IntraDcOnlyWedgeEnabledFlag : InterDcOnlyEnabledFlag) && PartMode == PART_2Nx2N)	
dc_only_flag[x0][y0]	ae(v)
}	
if(CuPredMode[x0][y0] != MODE_INTRA && PartMode == PART_2Nx2N) {	
if(IvResPredEnabledFlag && RpRefPicAvailFlag)	
iv_res_pred_weight_idx[x0][y0]	ae(v)
if(slice_ic_enabled_flag && icCuEnableFlag && iv_res_pred_weight_idx[x0][y0] == 0)	
illu_comp_flag[x0][y0]	ae(v)
}	
}	
}	

I.7.3.8.5.3 Depth DCs syntax

depth_dcs(x0, y0, log2CbSize) {	**Descriptor**
nCbS = (1 << log2CbSize)	
pbOffset = (PartMode == PART_NxN && CuPredMode[x0][y0] == MODE_INTRA) ? (nCbS / 2) : nCbS	
for(j = 0; j < nCbS; j = j + pbOffset)	
for(k = 0; k < nCbS; k = k + pbOffset)	
if(DimFlag[x0 + k][y0 + j] || DcOnlyFlag[x0][y0]) {	
if(CuPredMode[x0][y0] == MODE_INTRA && DcOnlyFlag[x0][y0])	
depth_dc_present_flag[x0 + k][y0 + j]	ae(v)
dcNumSeg = DimFlag[x0 + k][y0 + j] ? 2 : 1	
if(depth_dc_present_flag[x0 + k][y0 + j])	
for(i = 0; i < dcNumSeg; i++) {	
depth_dc_abs[x0 + k][y0 + j][i]	ae(v)
if((depth_dc_abs[x0 + k][y0 + j][i] − dcNumSeg + 2) > 0)	
depth_dc_sign_flag[x0 + k][y0 + j][i]	ae(v)
}	
}	
}	

I.7.3.8.6 Prediction unit syntax

The specifications in clause F.7.3.8.6 apply.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

I.7.3.8.7 PCM sample syntax

The specifications in clause F.7.3.8.7 apply.

I.7.3.8.8 Transform tree syntax

The specifications in clause F.7.3.8.8 apply.

I.7.3.8.9 Motion vector difference coding syntax

The specifications in clause F.7.3.8.9 apply.

I.7.3.8.10 Transform unit syntax

The specifications in clause F.7.3.8.10 apply.

I.7.3.8.11 Residual coding syntax

The specifications in clause F.7.3.8.11 apply.

I.7.3.8.12 Cross-component prediction syntax

The specifications in clause F.7.3.8.12 apply.

I.7.4 Semantics

I.7.4.1 General

I.7.4.2 NAL unit semantics

I.7.4.2.1 General NAL unit semantics

The specifications in clause F.7.4.2.1 apply.

I.7.4.2.2 NAL unit header semantics

The specifications in clause F.7.4.2.2 apply.

I.7.4.2.3 Encapsulation of an SODB within an RBSP (informative)

The specifications in clause F.7.4.2.3 apply.

I.7.4.2.4 Order of NAL units and association to coded pictures, access units, and coded video sequences

The specifications in clause F.7.4.2.4 and all its subclauses apply.

I.7.4.3 Raw byte sequence payloads, trailing bits, and byte alignment semantics

I.7.4.3.1 Video parameter set RBSP semantics

The specifications in clause F.7.4.3.1 apply with the following modifications and additions:

vps_extension2_flag equal to 0 specifies that no vps_3d_extension( ) syntax structure and no vps_extension_data_flag syntax elements are present in the VPS RBSP syntax structure. vps_extension2_flag equal to 1 specifies that the vps_3d_extension( ) syntax structure and vps_extension_data_flag syntax elements may be present in the VPS RBSP syntax structure. When MaxLayersMinus1 is greater than 0, vps_extension2_flag shall be equal to 1.

vps_3d_extension_flag equal to 0 specifies that no vps_3d_extension( ) syntax structure is present in the VPS RBSP syntax structure. vps_3d_extension_flag equal to 1 specifies that the vps_3d_extension( ) syntax structure is present in the VPS RBSP syntax structure. When MaxLayersMinus1 is greater than 0, vps_3d_extension_flag shall be equal to 1.

vps_3d_extension_alignment_bit_equal_to_one shall be equal to 1.

vps_extension3_flag equal to 0 specifies that no vps_extension_data_flag syntax elements are present in the VPS RBSP syntax structure. vps_extension3_flag shall be equal to 0 in bitstreams conforming to this version of this Specification. The value of 1 for vps_extension3_flag is reserved for future use by ITU-T | ISO/IEC. Decoders conforming to this version of this Specification shall ignore all data that follow the value 1 for vps_extension3_flag in a VPS RBSP.

vps_extension_data_flag may have any value. Its presence and value do not affect decoder conformance to profiles specified in Annexes A, G, H, or I. Decoders conforming to a profile specified in Annexes A, G, H, or I shall ignore all vps_extension_data_flag syntax elements.

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
Documents
Document Sharing Hub

I.7.4.3.1.1 Video parameter set extension semantics

The specifications in clause F.7.4.3.1.1 apply with the following additions and modifications:

direct_dependency_type[i][j] indicates the type of dependency between the layer predLayer with nuh_layer_id equal layer_id_in_nuh[i] and the layer refLayer with nuh_layer_id equal to layer_id_in_nuh[j]. ((direct_dependency_type[i][j] +1) & 0x1) greater than 0 specifies that samples of refLayer may be used for inter-layer prediction of predLayer. ((direct_dependency_type[i][j] +1) & 0x1) equal to 0 specifies that samples of refLayer are not used for inter-layer prediction of predLayer. ((direct_dependency_type[i][j] +1) & 0x2) greater than 0 specifies that motion vectors of refLayer may be used for inter-layer prediction of predLayer. ((direct_dependency_type[i][j] +1) & 0x2) equal to 0 specifies that motion vectors of refLayer are not used for inter-layer prediction of predLayer. ((direct_dependency_type[i][j] +1) & 0x4) greater than 0 specifies that coding quadtree and coding unit partitioning information of refLayer may be used for inter-layer prediction of predLayer. ((direct_dependency_type[i][j] +1) & 0x4) equal to 0 specifies that coding quadtree and coding unit partitioning information of refLayer are not used for inter-layer prediction of the predLayer.

The length of the direct_dependency_type[i][j] syntax element is direct_dep_type_len_minus2 + 2 bits. Although the value of direct_dependency_type[i][j] shall be in the range of 0 to 2, inclusive, when predLayer conforms to a profile specified in Annexes A, G, or H, and in the range of 0 to 6, inclusive, when the predLayer conforms to a profile specified in Annex I, decoders shall allow values of direct_dependency_type[i][j] in the range of 0 to $2^{32} - 2$, inclusive, to appear in the syntax.

The list ViewOIdxList[idx] is derived as follows:

```
idx = 0
ViewOIdxList[ idx++ ] = 0
for( i = 1; i <= MaxLayersMinus1; i++ ) {
    newViewFlag = 1
    for( j = 0; j < i; j++ )                                                    (I−7)
        if( ViewOrderIdx[ layer_id_in_nuh[ i ] ] == ViewOrderIdx[ layer_id_in_nuh[ j ] ] )
            newViewFlag = 0
    if( newViewFlag )
        ViewOIdxList[ idx++ ] = ViewOrderIdx[ lId ]
}
```

The variables NumRefListLayers[iNuhLId] and IdRefListLayer[iNuhLId] are derived as follows:

```
for( i = 0; i <= MaxLayersMinus1; i++ ) {
    iNuhLId = layer_id_in_nuh[ i ]
    NumRefListLayers[ iNuhLId ] = 0
    for( j = 0; j < NumDirectRefLayers[ iNuhLId ]; j++ ) {
        jNuhLId = IdDirectRefLayer[ iNuhLId ][ j ]                              (I−8)
        if( DepthLayerFlag[ iNuhLId ] == DepthLayerFlag[ jNuhLId ] )
            IdRefListLayer[ iNuhLId ][ NumRefListLayers[ iNuhLId ]++ ] = jNuhLId
    }
}
```

The variables ViewCompLayerPresentFlag[iViewOIdx][depFlag] and ViewCompLayerId[iViewOIdx][depFlag] are derived as follows:

```
for( depFlag = 0; depFlag <= 1; depFlag++ )
    for( i = 0; i < NumViews; i++ ) {
        iViewOIdx = ViewOIdxList[ i ]
        layerId = −1
        for( j = 0; j <= MaxLayersMinus1; j++ ) {
            jNuhLId = layer_id_in_nuh[ j ]                                      (I−9)
            if( DepthLayerFlag[ jNuhLId ] == depFlag && ViewOrderIdx[ jNuhLId ] == iViewOIdx
                && DependencyId[ jNuhLId ] == 0 && AuxId[ jNuhLId ] == 0 )
                layerId = jNuhLId
        }
        ViewCompLayerPresentFlag[ iViewOIdx ][ depFlag ] = ( layerId != −1 )
        ViewCompLayerId[ iViewOIdx ][ depFlag ] = layerId
    }
```

The function ViewIdx(picX) is specified as follows:

```
ViewIdx( picX ) = ViewIdx of the picture picX                                   (I−10)
```


Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

The function ViewIdVal(picX) is specified as follows:

ViewIdVal(picX) = view_id_val[ViewIdx(picX)] (I−11)

I.7.4.3.1.2 Representation format semantics

The specifications in clause F.7.4.3.1.2 apply.

I.7.4.3.1.3 DPB size semantics

The specifications in clause F.7.4.3.1.3 apply.

I.7.4.3.1.4 VPS VUI semantics

The specifications in clause F.7.4.3.1.4 apply.

I.7.4.3.1.5 Video signal info semantics

The specifications in clause F.7.4.3.1.5 apply.

I.7.4.3.1.6 VPS VUI bitstream partition HRD parameters semantics

The specifications in clause F.7.4.3.1.6 apply.

I.7.4.3.1.7 Video parameter set 3D extension semantics

cp_precision + $BitDepth_Y - 1$ specifies the precision of the vps_cp_scale[i][j] and vps_cp_inv_scale_plus_scale[i][j] syntax elements present in the VPS and the cp_scale[j] and cp_inv_scale_plus_scale[j] syntax elements present in slice headers. The value of cp_precision shall be in the range of 0 to 5, inclusive.

num_cp[i], when cp_in_slice_segment_header_flag[i] is equal to 0, specifies the number of vps_cp_scale[i][j], vps_cp_off[i][j], vps_cp_inv_scale_plus_scale[i][j], and vps_cp_inv_off_plus_off[i][j] syntax elements present for the view with ViewIdx equal to i in the VPS. num_cp[i], when cp_in_slice_segment_header_flag[i] is equal to 1, specifies the number of cp_scale[j], cp_off[j], cp_inv_scale_plus_scale[j], and cp_inv_off_plus_off[j] syntax elements present in slice headers of layers with ViewIdx equal to i.

cp_in_slice_segment_header_flag[i] equal to 1 specifies that the syntax elements vps_cp_scale[i][j], vps_cp_off[i][j], vps_cp_inv_scale_plus_scale[i][j], and vps_cp_inv_off_plus_off[i][j] for the view with ViewIdx equal to i are not present in the VPS and that the syntax elements cp_scale[j], cp_off[j], cp_inv_scale_plus_scale[j], and cp_inv_off_plus_off[j] may be present in slice headers of layers with ViewIdx equal to i. cp_in_slice_segment_header_flag[i] equal to 0 specifies that the vps_cp_scale[i][j], vps_cp_off[i][j], vps_cp_inv_scale_plus_scale[i][j], and vps_cp_inv_off_plus_off[i][j] syntax elements for the view with ViewIdx equal to i are present in the VPS and that the syntax elements cp_scale[j], cp_off[j], cp_inv_scale_plus_scale[j], and cp_inv_off_plus_off[j] are not present in slice headers of layers with ViewIdx equal to i. When not present, the value of cp_in_slice_segment_header_flag[i] is inferred to be equal to 0.

cp_ref_voi[i][m], when cp_in_slice_segment_header_flag[i] is equal to 0, specifies the ViewIdx value j of the view to which the m-th vps_cp_scale[i][j], vps_cp_off[i][j], vps_cp_inv_scale_plus_scale[i][j], and vps_cp_inv_off_plus_off[i][j] syntax element present for the view with ViewIdx equal to i in the VPS is related to. cp_ref_voi[i][m], when cp_in_slice_segment_header_flag[i] is equal to 1, specifies the ViewIdx value j of the view to which the m-th cp_scale[j], cp_off[j], cp_inv_scale_plus_scale[j], and cp_inv_off_plus_off[j] syntax element present in the slice headers of layers with ViewIdx equal to i is related to. The value of cp_ref_voi[i][m] shall be in the range of 0 to 65535, inclusive. It is a requirement of bitstream conformance that cp_ref_voi[i][x] is not equal to cp_ref_voi[i][y] for any values of x and y in the range of 0 to num_cp[i] − 1, inclusive, when x is not equal to y.

For n and m in the range of 0 to NumViews − 1, inclusive, the variable CpPresentFlag[ViewOIdxList[n]][ViewOIdxList[m]] is set equal to 0 and modified as follows:

```
for( n = 1; n < NumViews; n++ ) {
    i = ViewOIdxList[ n ]
    for( m = 0; m < num_cp[ i ]; m++ )                                  (I−12)
        CpPresentFlag[ i ][ cp_ref_voi[ i ][ m ] ] = 1
}
```

vps_cp_scale[i][j], **vps_cp_off**[i][j], **vps_cp_inv_scale_plus_scale**[i][j], and **vps_cp_inv_off_plus_off**[i][j] specify parameters for derivation of a horizontal component of a disparity vector from a depth value and may be used to infer the values of the cp_scale[j], cp_off[j], cp_inv_scale_plus_scale[j], and cp_inv_off_plus_off[j] syntax elements in slice headers of layers with ViewIdx equal to i. When both a texture layer and a depth layer with ViewIdx equal to i are present, the conversion parameters are associated with the texture layer with ViewIdx equal to i.

AVA Documents
Document Sharing Hub

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015

I.7.4.3.2 Sequence parameter set RBSP semantics

I.7.4.3.2.1 General sequence parameter set RBSP semantics

The specifications in clause F.7.4.3.2.1 apply.

I.7.4.3.2.2 Sequence parameter set range extension semantics

The specifications in clause F.7.4.3.2.2 apply.

I.7.4.3.2.3 Sequence parameter set multilayer extension semantic

The specifications in clause F.7.4.3.2.3 apply.

I.7.4.3.2.4 Sequence parameter set 3D extension semantics

iv_di_mc_enabled_flag[d] equal to 1 specifies that the derivation process for inter-view predicted merging candidates and the derivation process for disparity information merging candidates may be used in the decoding process of layers with DepthFlag equal to d. iv_di_mc_enabled_flag[d] equal to 0 specifies that derivation process for inter-view predicted merging candidates and the derivation process for disparity information merging candidates is not used in the decoding process of layers with DepthFlag equal to d. When not present, the value of iv_di_mc_enabled_flag[d] is inferred to be equal to 0.

iv_mv_scal_enabled_flag[d] equal to 1 specifies that motion vectors used for inter-view prediction may be scaled based on view_id_val values in the decoding process of layers with DepthFlag equal to d. iv_mv_scal_enabled_flag[d] equal to 0 specifies that motion vectors used for inter-view prediction are not scaled based on view_id_val values in the decoding process of layers with DepthFlag equal to d. When not present, the value of iv_mv_scal_enabled_flag[d] is inferred to be equal to 0.

log2_ivmc_sub_pb_size_minus3[d], when iv_di_mc_enabled_flag[d] is equal to 1 and d is equal to 0, is used to derive the minimum size of sub-block partitions used in the derivation process for sub-block partition motion vectors for an inter-layer predicted merging candidate in the decoding process of layers with DepthFlag equal to d. When not present, the value of log2_ivmc_sub_pb_size_minus3[d] is inferred to be equal to (CtbLog2SizeY − 3). The value of log2_ivmc_sub_pb_size_minus3[d] shall be in the range of (MinCbLog2SizeY − 3) to (CtbLog2SizeY − 3), inclusive.

iv_res_pred_enabled_flag[d] equal to 1 specifies that the iv_res_pred_weight_idx syntax element may be present in coding units of layers with DepthFlag equal to d. iv_res_pred_enabled_flag[d] equal to 0 specifies that the iv_res_pred_weight_idx syntax element is not present coding units of layers with DepthFlag equal to d. When not present, the value of iv_res_pred_enabled_flag[d] is inferred to be equal to 0.

vsp_mc_enabled_flag[d] equal to 1 specifies that the derivation process for a view synthesis prediction merging candidate may be used in the decoding process of layers with DepthFlag equal to d. vsp_mc_enabled_flag[d] equal to 0 specifies that the derivation process for a view synthesis prediction merging candidate is not used the decoding process of layers with DepthFlag equal to d. When not present, the value of vsp_mc_enabled_flag[d] is inferred to be equal to 0.

dbbp_enabled_flag[d] equal to 1 specifies that the dbbp_flag syntax element may be present in coding units of layers with DepthFlag equal to d. dbbp_enabled_flag[d] equal to 0 specifies that the dbbp_flag syntax element is not present coding units of layers with DepthFlag equal to d. When not present, the value of dbbp_enabled_flag[d] is inferred to be equal to 0.

depth_ref_enabled_flag[d] equal to 1 specifies that the derivation process for a depth or disparity sample array from a depth picture may be used in the derivation process for a disparity vector for texture layers in the decoding process of layers with DepthFlag equal to d. depth_ref_enabled_flag[d] equal to 0 specifies that derivation process for a depth or disparity sample array from a depth picture is not used in the derivation process for a disparity vector for texture layers in the decoding process of layers with DepthFlag equal to d. When not present, the value of depth_ref_enabled_flag[d] is inferred to be equal to 0.

tex_mc_enabled_flag[d] equal to 1 specifies that the derivation process for motion vectors for the texture merge candidate may be used in the decoding process of layers with DepthFlag equal to d. tex_mc_enabled_flag[d] equal to 0 specifies that the derivation process for motion vectors for the texture merge candidate is not used in the decoding process of layers with DepthFlag equal to d. When not present, the value of tex_mc_enabled_flag[d] is inferred to be equal to 0.

log2_texmc_sub_pb_size_minus3[d], when tex_mc_enabled_flag[d] is equal to 1, is used to derive the minimum size of sub-block partitions used in the derivation process for sub-block partition motion vectors for an inter-layer predicted merging candidate in the decoding process of layers with DepthFlag equal to d. The value of log2_texmc_sub_pb_size_minus3[layerId] shall be in the range of (MinCbLog2SizeY − 3) to (CtbLog2SizeY − 3), inclusive.

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved

intra_contour_enabled_flag[d] equal to 1 specifies that the intra prediction mode INTRA_CONTOUR using depth intra contour prediction may be used in the decoding process of layers with DepthFlag equal to d. intra_contour_enabled_flag[d] equal to 0 specifies that the intra prediction mode INTRA_CONTOUR using depth intra contour prediction is not used in the decoding process of layers with DepthFlag equal to d. When not present, intra_contour_enabled_flag[d] is inferred to be equal to 0.

intra_dc_only_wedge_enabled_flag[d] equal to 1 specifies that the dc_only_flag syntax element may be present in coding units coded in an intra prediction mode of layers with DepthFlag equal to d, and that the intra prediction mode INTRA_WEDGE may be used in the decoding process of layers with DepthFlag equal to d. intra_dc_only_wedge_enabled_flag[d] equal to 0 specifies that the dc_only_flag syntax element is not present in coding units coded in an intra prediction mode of layers with DepthFlag equal to d and that the intra prediction mode INTRA_WEDGE is not used in the decoding process of layers with DepthFlag equal to d. When not present, the value of intra_dc_only_wedge_enabled_flag[d] is inferred to be equal to 0.

cqt_cu_part_pred_enabled_flag[d] equal to 1 specifies that coding quadtree and coding unit partitioning information may be inter-component predicted in the decoding process of layers with DepthFlag equal to d. cqt_cu_part_pred_enabled_flag[d] equal to 0 specifies that coding quadtree and coding unit partitioning information are not inter-component predicted in the decoding process of layers with DepthFlag equal to d. When not present, the value of cqt_cu_part_pred_enabled_flag[d] is inferred to be equal to 0.

inter_dc_only_enabled_flag[d] equal to 1 specifies that the dc_only_flag syntax element may be present in coding units coded an in inter prediction mode of layers with DepthFlag equal to d. inter_dc_only_enabled_flag[d] equal to 0 specifies that the dc_only_flag syntax element is not present in coding units coded in an inter prediction mode of layers with DepthFlag equal to d. When not present, the value of inter_dc_only_enabled_flag[layerId] is inferred to be equal to 0.

skip_intra_enabled_flag[d] equal to 1 specifies that the skip_intra_flag syntax element may be present in coding units of layers with DepthFlag equal to d. skip_intra_enabled_flag[d] equal to 0 specifies that the skip_intra_flag syntax element is not present in coding units of layers with DepthFlag equal to d. When not present, the value of skip_intra_enabled_flag[layerId] is inferred to be equal to 0.

I.7.4.3.3 Picture parameter set RBSP semantics

I.7.4.3.3.1 General picture parameter set RBSP semantics

The specifications in clause F.7.4.3.3.1 apply.

I.7.4.3.3.2 Picture parameter set range extension semantics

The specifications in clause F.7.4.3.3.2 apply.

I.7.4.3.3.3 Picture parameter set multilayer extension semantics

The specifications in clause F.7.4.3.3.3 apply.

I.7.4.3.3.4 General colour mapping table semantics

The specifications in clause F.7.4.3.3.4 apply.

I.7.4.3.3.5 Colour mapping octants semantics

The specifications in clause F.7.4.3.3.5 apply.

I.7.4.3.3.6 Picture parameter set 3D extension semantics

dlts_present_flag equal to 1 specifies that syntax elements for the derivation of depth look-up tables are present in the PPS. dlts_present_flag equal to 0 specifies that syntax elements for the derivation of depth look-up tables are not present in the PPS.

The variables NumDepthLayers and DepIdxToLId[j] are derived as follows:

```
j = 0
for( i = 0; i  <=  MaxNumLayersMinus1; i++ ) {
    layerId = layer_id_in_nuh[ i ]
    if( DepthLayerFlag[ layerId ] )                                    (I−13)
        DepIdxToLId[ j++ ] = layerId
}
NumDepthLayers = j
```

pps_depth_layers_minus1 plus 1 specifies the number of depth layers. pps_depth_layers_minus1 shall be equal to

© ISO/IEC 2015 – All rights reserved
Full document details: ISO/IEC 23008-2:2015/Amd 1:2015
AVA Documents Document Sharing Hub

NumDepthLayers − 1.

pps_bit_depth_for_depth_layers_minus8 plus 8 specifies the bit depth of the samples in depth layers. It is a requirement of bitstream conformance that pps_bit_depth_for_depth_layers_minus8 shall be equal to bit_depth_luma_minus8 of the SPS the current PPS refers to.

The variable depthMaxValue is set equal to (1 << (pps_bit_depth_for_depth_layers_minus8 + 8)) − 1.

dlt_flag[i] equal to 1 specifies that the a depth look-up table for the layer with nuh_layer_id equal to DepIdxToLId[i] is present in the PPS and used for the decoding of the layer with nuh_layer_id equal to DepIdxToLId[i]. dlt_flag[i] equal to 0 specifies that a depth look-up table is not present for the layer with nuh_layer_id equal to DepIdxToLId[i]. When not present, the value of dlt_flag[i] is inferred to be equal to 0.

For i in the range of 0 to NumDepthLayers − 1, inclusive, the variable DltFlag[DepIdxToLId[i]] is set equal to dlt_flag[i].

dlt_pred_flag[i] equal to 1 indicates that the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i] is predicted from the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[0]. dlt_pred_flag[i] equal to 0 indicates that the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i] is not predicted from any other depth look-up table. The value of dlt_pred_flag[0] shall be equal to 0. It is a requirement of bitstream conformance that, when dlt_flag[0] is equal to 0, dlt_pred_flag[i] shall be equal to 0.

dlt_val_flags_present_flag[i] equal to 1 specifies the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i] is derived from dlt_value_flag[i][j] syntax elements. dlt_val_flags_present_flag[i] equal to 0 specifies the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i] is derived from the delta_dlt( ) syntax structure. When not present, the value of dlt_val_flags_present_flag[i] is inferred to be equal to 0.

dlt_value_flag[i][j] equal to 1 specifies that j is an entry in the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i]. dlt_value_flag[i][j] equal to 0 specifies that j is not an entry in the depth look-up table of the layer with nuh_layer_id equal to DepIdxToLId[i].

When dlt_val_flags_present_flag[i] is equal to 1, the following applies:

– The variable layerId is set equal to DepIdxToLId[i].

– The variables DltVal[layerId][n] and NumValDlt[layerId] of the depth look-up table of the layer with nuh_layer_id equal to layerId are derived as follows:

```
for( n = 0, j = 0; j <= depthMaxValue; j++ )
    if( dlt_value_flag[ i ][ j ] )
        DltVal[ layerId ][ n++ ] = j
NumValDlt[ layerId ] = n
```
(I−14)

I.7.4.3.3.7 Delta depth look-up table semantics

num_val_delta_dlt specifies the number of elements in the list deltaList. The length of num_val_delta_dlt syntax element is pps_bit_depth_for_depth_layers_minus8 + 8 bits.

max_diff specifies the maximum difference between two consecutive elements in the list deltaList. The length of max_diff syntax element is pps_bit_depth_for_depth_layers_minus8 + 8 bits. When not present, the value of max_diff is inferred to be equal to 0.

min_diff_minus1 specifies the minimum difference between two consecutive elements in the list deltaList. min_diff_minus1 shall be in the range of 0 to max_diff − 1, inclusive. The length of the min_diff_minus1 syntax element is Ceil(Log2(max_diff + 1)) bits. When not present, the value of min_diff_minus1 is inferred to be equal to (max_diff − 1).

The variable minDiff is set equal to (min_diff_minus1 + 1).

delta_dlt_val0 specifies the 0-th element in the list deltaList. The length of the delta_dlt_val0 syntax element is pps_bit_depth_for_depth_layers_minus8 + 8 bits.

delta_val_diff_minus_min[k] plus minDiff specifies the difference between the k-th element and the (k − 1)-th element in the list deltaList. The length of delta_val_diff_minus_min[k] syntax element is Ceil(Log2(max_diff − minDiff + 1)) bits. When not present, the value of delta_val_diff_minus_min[k] is inferred to be equal to 0.

The list deltaList is derived as follows:

```
deltaList[ 0 ] = delta_dlt_val0
for( k = 1; k < num_val_delta_dlt; k++ )
```
(I−15)

Documents
Document Sharing Hub
Full document details: ISO_IEC 23008-2_2015_Amd 1_2015
© ISO/IEC 2015 – All rights reserved