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

**Part 2:
High efficiency video coding**

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

Partie 2: Codage vidéo à haute efficacité



Reference number
ISO/IEC 23008-2:2023(E)

Documents
Document Sharing Hub
© ISO/IEC 2023



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2023

All rights reserved. Unless otherwise specified, or required in the context of its implementation, 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
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents	Page
Foreword	vii
Introduction	viii
1 Scope	1
2 Normative references	1
3 Terms and definitions	1
4 Abbreviated terms	22
5 Conventions	24
5.1 General.....	24
5.2 Arithmetic operators.....	24
5.3 Logical operators	25
5.4 Relational operators.....	25
5.5 Bit-wise operators.....	25
5.6 Assignment operators.....	26
5.7 Range notation.....	26
5.8 Mathematical functions.....	26
5.9 Order of operation precedence.....	27
5.10 Variables, syntax elements, and tables	28
5.11 Text description of logical operations	29
5.12 Processes.....	30
6 Bitstream and picture formats, partitionings, scanning processes, and neighbouring relationships	31
6.1 Bitstream formats.....	31
6.2 Source, decoded, and output picture formats.....	31
6.3 Partitioning of pictures, slices, slice segments, tiles, CTUs, and CTBs	34
6.3.1 Partitioning of pictures into slices, slice segments, and tiles	34
6.3.2 Block and quadtree structures	36
6.3.3 Spatial or component-wise partitionings	37
6.4 Availability processes	37
6.4.1 Derivation process for z-scan order block availability	37
6.4.2 Derivation process for prediction block availability.....	38
6.5 Scanning processes	40
6.5.1 CTB raster and tile scanning conversion process	40
6.5.2 Z-scan order array initialization process.....	41
6.5.3 Up-right diagonal scan order array initialization process.....	41
6.5.4 Horizontal scan order array initialization process.....	42
6.5.5 Vertical scan order array initialization process.....	42

6.5.6	Traverse scan order array initialization process	43
7	Syntax and semantics	43
7.1	Method of specifying syntax in tabular form	43
7.2	Specification of syntax functions and descriptors	44
7.3	Syntax in tabular form	46
7.3.1	NAL unit syntax	46
7.3.2	Raw byte sequence payloads, trailing bits, and byte alignment syntax	47
7.3.3	Profile, tier and level syntax	56
7.3.4	Scaling list data syntax	59
7.3.5	Supplemental enhancement information message syntax	60
7.3.6	Slice segment header syntax	60
7.3.7	Short-term reference picture set syntax	65
7.3.8	Slice segment data syntax	66
7.4	Semantics	81
7.4.1	General	81
7.4.2	NAL unit semantics	81
7.4.3	Raw byte sequence payloads, trailing bits, and byte alignment semantics	92
7.4.4	Profile, tier, and level semantics	114
7.4.5	Scaling list data semantics	119
7.4.6	Supplemental enhancement information message semantics	121
7.4.7	Slice segment header semantics	122
7.4.8	Short-term reference picture set semantics	131
7.4.9	Slice segment data semantics	134
8	Decoding process	151
8.1	General decoding process	151
8.1.1	General	151
8.1.2	CVSG decoding process	151
8.1.3	Decoding process for a coded picture with nuh_layer_id equal to 0	152
8.2	NAL unit decoding process	154
8.3	Slice decoding process	154
8.3.1	Decoding process for picture order count	154
8.3.2	Decoding process for reference picture set	155
8.3.3	Decoding process for generating unavailable reference pictures	161
8.3.4	Decoding process for reference picture lists construction	162
8.3.5	Decoding process for collocated picture and no backward prediction flag	163
8.4	Decoding process for coding units coded in intra prediction mode	164
8.4.1	General decoding process for coding units coded in intra prediction mode	164

8.4.2	Derivation process for luma intra prediction mode.....	168
8.4.3	Derivation process for chroma intra prediction mode	171
8.4.4	Decoding process for intra blocks	171
8.5	Decoding process for coding units coded in inter prediction mode.....	184
8.5.1	General decoding process for coding units coded in inter prediction mode	184
8.5.2	Inter prediction process.....	185
8.5.3	Decoding process for prediction units in inter prediction mode.....	188
8.5.4	Decoding process for the residual signal of coding units coded in inter prediction mode	221
8.6	Scaling, transformation and array construction process prior to deblocking filter process	226
8.6.1	Derivation process for quantization parameters	226
8.6.2	Scaling and transformation process	228
8.6.3	Scaling process for transform coefficients	230
8.6.4	Transformation process for scaled transform coefficients	231
8.6.5	Residual modification process for blocks using a transform bypass	234
8.6.6	Residual modification process for transform blocks using cross-component prediction.....	235
8.6.7	Picture construction process prior to in-loop filter process	235
8.6.8	Residual modification process for blocks using adaptive colour transform.....	236
8.7	In-loop filter process	238
8.7.1	General	238
8.7.2	Deblocking filter process.....	239
8.7.3	Sample adaptive offset process.....	257
9	Parsing process.....	260
9.1	General	260
9.2	Parsing process for 0-th order Exp-Golomb codes	260
9.2.1	General	260
9.2.2	Mapping process for signed Exp-Golomb codes	262
9.3	CABAC parsing process for slice segment data	262
9.3.1	General	262
9.3.2	Initialization process.....	265
9.3.3	Binarization process.....	278
9.3.4	Decoding process flow.....	288
9.3.5	Arithmetic encoding process	304

10 Sub-bitstream extraction process	311
Annex A (normative) Profiles, tiers and levels	313
Annex B (normative) Byte stream format	343
Annex C (normative) Hypothetical reference decoder	346
Annex D (normative) Supplemental enhancement information	367
Annex E (normative) Video usability information	529
Annex F (normative) Common specifications for multi-layer extensions	559
Annex G (normative) Multiview high efficiency video coding	717
Annex H (normative) Scalable high efficiency video coding	743
Annex I (normative) 3D high efficiency video coding	774
Bibliography	898

Foreword

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

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and <https://patents.iec.ch>. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*, in collaboration with ITU-T (as Rec. ITU-T H.265).

This fifth edition cancels and replaces the fourth edition (ISO/IEC 23008-2:2020), which has been technically revised. It also incorporates the amendment ISO/IEC 23008-2:2020/Amd 1:2021.

The main changes are as follows:

- the specification of four additional “levels” of high capability for all profiles, referred to as levels 6.3, 7, 7.1, and 7.2;
- the specification of level 8.5 for the video profiles, which had previously been specified only for still picture profiles and which provides a way to identify bitstreams for which it is possible that the bitstreams do not conform to non-syntactical constraints specified for bitstream conformance to other levels.

A list of all parts in the ISO/IEC 23008 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

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 document was developed in the JCT-VC and the JCT-3V.

Purpose

This document 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 communications. It is also designed to enable the use of the coded video representation in a flexible manner for a wide variety of network environments as well as to enable the use of multi-core parallel encoding and decoding devices. The use of this document 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. Supports for higher bit depths and enhanced chroma formats, including the use of full-resolution chroma are provided. Support for scalability enables video transmission on networks with varying transmission conditions and other scenarios involving multiple bit rate services. Support for multiview enables representation of video content with multiple camera views and optional auxiliary information. Support for 3D enables joint representation of video content and depth information with multiple camera views.

Applications

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

- broadcast (cable TV on optical networks / copper, satellite, terrestrial, etc.);
- camcorders;
- content production and distribution;
- digital cinema;
- home cinema;
- internet streaming, download and play;
- medical imaging;
- mobile streaming, broadcast and communications;
- real-time conversational services (videoconferencing, videophone, telepresence, etc.);
- remote video surveillance;
- storage media (optical disks, digital video tape recorder, etc.);

— wireless display.

Publication and versions of this document

This document has been jointly developed by ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). It is published as technically-aligned twin text in both ITU-T and ISO/IEC. As the basis text has been drafted to become both an ITU-T Recommendation and an ISO/IEC International Standard, the term "Specification" (with capitalization to indicate that it refers to the whole of the text) is used herein when the text refers to itself.

This is the fifth version of this document and the fourth edition published by ISO/IEC.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 1 refers to the first approved version of this document. The first edition published by ISO/IEC as ISO/IEC 23008-2:2013 corresponded to the first version.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 2 refers to the integrated text additionally containing format range extensions, scalability extensions, multiview extensions, additional supplement enhancement information, and corrections to various minor defects in the prior content of the specification. The second edition published by ISO/IEC as ISO/IEC 23008-2:2015 corresponded to the second version.

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

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 4 refers to the integrated text additionally containing screen content coding extensions profiles, scalable range extensions profiles, additional high throughput profiles, additional supplement enhancement information, additional colour representation identifiers, and corrections to various minor defects in the prior content of the specification. The third edition published by ISO/IEC as ISO/IEC 23008-2:2017 corresponded to the fourth version.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 5 refers to the integrated text additionally containing additional SEI messages that include omnidirectional video specific SEI messages, a Monochrome 10 profile, a Main 10 Still Picture profile, and corrections to various minor defects in the prior content of the specification.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 6 refers to the integrated text additionally containing additional SEI messages for SEI manifest and SEI prefix, and corrections to various minor defects in the prior content of the specification.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 7 refers to the integrated text additionally containing the fisheye video information SEI message and the annotated regions SEI message, and corrections to various minor defects in the prior content of the specification. The fourth edition published by ISO/IEC as ISO/IEC 23008-2:2020 corresponded to the seventh version.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 8 refers to the integrated text additionally containing the shutter interval SEI message and corrections to various minor defects in the prior content of the specification.

Rec. ITU-T H.265 | ISO/IEC 23008-2 version 9 (the current version) refers to the integrated text additionally containing specification of levels 6.3, 7, 7.1, and 7.2, the specification of level 8.5 for the video profiles, and corrections to various minor defects in the prior content of the specification. This fifth edition published by ISO/IEC as ISO/IEC 23008-2:202x corresponds to the ninth version.

Profiles, tiers, and levels

This document 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 document, 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 document will facilitate video data interchange among different applications.

Considering the practicality of implementing the full syntax of this document, however, a limited number of subsets of the syntax are also stipulated by means of "profiles", "tiers", 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 in this document. 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 economical 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, "tiers" and "levels" are specified within each profile. A level of a tier 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). A level specified for a lower tier is more constrained than a level specified for a higher tier.

Coded video content conforming to this document 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.

Overview of the design characteristics

The coded representation specified in the syntax is designed to enable a high compression capability for a desired image or video quality. 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 may then be further compressed using a transform to remove spatial correlation inside the transform block before it is quantized, producing a possibly 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 may also be further compressed using a variety of prediction mechanisms, and, after prediction, are combined with the quantized transform coefficient information and encoded using arithmetic coding.

How to read this document

It is suggested that the reader starts with Clause 1 (Scope) and moves on to Clause 3 (Terms and 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 7.1 to 7.3 for syntactical order and see 7.4 for semantics; e.g. 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). Clause 10 (Sub-bitstream extraction process) specifies the sub-bitstream extraction process. Finally, Clause 8 (Decoding process) specifies how the syntax elements are mapped into decoded samples. Throughout reading this document, the reader should refer to Clauses 2 (Normative references), 4 (Abbreviated terms), and 5 (Conventions) as needed. Annexes A through I also form an integral part of this document.

Annex A specifies profiles each being tailored to certain application domains, and defines the so-called tiers and 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, bitstream conformance, decoder conformance, and the use of the hypothetical reference decoder to check bitstream and decoder conformance. Annex D specifies syntax and semantics for supplemental enhancement information message payloads. Annex E specifies syntax and semantics of the video usability information parameters of the sequence parameter set. Annex F specifies general multi-layer support for bitstreams and decoders. Annex G contains support for multiview coding. Annex H contains support for scalability. Annex I contains support for 3D coding.

Throughout this document, statements appearing with the preamble "NOTE" are informative and are not an integral part of this document.

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

Part 2: High efficiency video coding

1 Scope

This document specifies high efficiency video coding.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 11664-1, *Colorimetry — Part 1: CIE standard colorimetric observers*

ISO 12232, *Photography — Digital still cameras — Determination of exposure index, ISO speed ratings, standard output sensitivity, and recommended exposure index*

ISO/IEC 10646, *Information technology — Universal Coded Character Set (UCS)*

ISO/IEC 11578, *Information technology — Open Systems Interconnection — Remote Procedure Call (RPC)*

ISO/IEC 23001-11, *Information technology — MPEG systems technologies — Part 11: Energy-efficient media consumption (green metadata)*

Recommendation ITU-T T.35, *Procedure for the allocation of ITU-T defined codes for non-standard facilities*

IETF RFC 1321, *The MD5 Message-Digest Algorithm*

IETF RFC 5646, *Tags for Identifying Languages*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <https://www.electropedia.org/>

3.1

access unit

set of *NAL units* (3.87) that are associated with each other according to a specified classification rule, are consecutive in *decoding order* (3.44), and contain exactly one *coded picture* (3.25) with *nuh_layer_id* equal to 0

Note 1 to entry: In addition to containing the *VCL NAL units* (3.181) of the coded picture with *nuh_layer_id* equal to 0, an access unit can also contain non-VCL NAL units. The decoding of an access unit with the decoding process specified in Clause 8 always results in a decoded picture with *nuh_layer_id* equal to 0.

Note 2 to entry: An access unit is defined differently in Annex F and might not contain a coded picture with *nuh_layer_id* equal to 0.

3.2

AC transform coefficient

transform coefficient (3.175) for which the *frequency index* (3.58) in at least one of the two dimensions is non-zero

3.3

associated IRAP picture

previous *IRAP picture* (3.71) in *decoding order* (3.44) (when present)

3.4

associated non-VCL NAL unit

non-VCL NAL unit (3.91) (when present) for a *VCL NAL unit* (3.180) where the VCL NAL unit is the *associated VCL NAL unit* (3.5) of the non-VCL NAL unit

3.5

associated VCL NAL unit

preceding *VCL NAL unit* (3.180) in *decoding order* (3.44) for a *non-VCL NAL unit* (3.91) with *nal_unit_type* equal to EOS_NUT, EOB_NUT, FD_NUT, or SUFFIX_SEI_NUT, or in the ranges of RSV_NVCL45..RSV_NVCL47 or UNSPEC56..UNSPEC63; or otherwise, the next VCL NAL unit in decoding order

3.6

azimuth circle

circle on a sphere connecting all points with the same azimuth value

Note 1 to entry: An azimuth circle is always a great circle like a longitude line on the earth.

3.7

base layer

layer in which all *NAL units* (3.87) have *nuh_layer_id* equal to 0

3.8

bin

one bit of a *bin string* (3.11)

3.9

binarization

set of *bin strings* (3.11) for all possible values of a *syntax element* (3.162)

3.10

binarization process

unique mapping process of all possible values of a *syntax element* (3.162) onto a set of *bin strings* (3.11)

3.11

bin string

intermediate binary representation of values of *syntax elements* (3.162) from the *binarization* (3.9) of the syntax element

3.12**bi-predictive slice****B slice**

slice (3.143) that is decoded using *intra prediction* (3.69) or using *inter prediction* (3.67) with at most two *motion vectors* (3.86) and *reference indices* (3.127) to predict the sample values of each *block* (3.14)

3.13**bitstream**

sequence of bits, in the form of a *NAL unit stream* (3.88) or a *byte stream* (3.21), that forms the representation of *coded pictures* (3.25) and associated data forming one or more *CVSs* (3.30)

3.14**block**

MxN (M-column by N-row) array of samples or an MxN array of *transform coefficients* (3.175)

3.15**broken link**

location in a *bitstream* (3.13) at which it is indicated that some subsequent *pictures* (3.99) in *decoding order* (3.44) may contain serious visual artefacts due to unspecified operations performed in the generation of the bitstream

3.16**broken link access access unit****BLA access unit**

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is a *BLA picture* (3.17)

3.17**broken link access picture****BLA picture**

IRAP picture (3.71) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to *BLA_W_LP*, *BLA_W_RADL*, or *BLA_N_LP*

Note 1 to entry: A BLA picture does not refer to any pictures other than itself for *inter prediction* (3.67) in its *decoding process* (3.45), and can be the first picture in the *bitstream* (3.13) in decoding order, or can appear later in the bitstream. Each BLA picture begins a new *CVS* (3.30), and has the same effect on the decoding process as an *IDR picture* (3.65). However, a BLA picture contains *syntax elements* (3.162) that specify a non-empty *RPS* (3.132). When a BLA picture for which each VCL NAL unit has *nal_unit_type* equal to *BLA_W_LP*, it can have associated RASL pictures, which are not output by the decoder and can be non-decodable, as they can contain references to pictures that are not present in the bitstream. When a BLA picture for which each VCL NAL unit has *nal_unit_type* equal to *BLA_W_LP*, it can also have associated RADL pictures, which are specified to be decoded. When a BLA picture for which each VCL NAL unit has *nal_unit_type* equal to *BLA_W_RADL*, it does not have associated RASL pictures but can have associated RADL pictures. When a BLA picture for which each VCL NAL unit has *nal_unit_type* equal to *BLA_N_LP*, it does not have any associated leading pictures.

3.18**buffering period**

set of *access units* (3.1) starting with an access unit that contains a buffering period SEI message and containing all subsequent access units in *decoding order* (3.44) up to but not including the next access unit (when present) that contains a buffering period SEI message

3.19**byte**

sequence of 8 bits, within which, when written or read as a sequence of bit values, the left-most and right-most bits represent the most and least significant bits, respectively

3.20

byte-aligned

position in a *bitstream* (3.13) is byte-aligned when the position is an integer multiple of 8 bits from the position of the first bit in the bitstream and a bit or *byte* (3.19) or *syntax element* (3.162) is said to be byte-aligned when the position at which it appears in a bitstream is byte-aligned

3.21

byte stream

sequence of bytes forming an encapsulation of a *NAL unit stream* (3.88) into a format containing *start code prefixes* (3.150) and *NAL units* (3.87)

Note 1 to entry: See Annex B.

3.22

chroma

Cb

Cr

sample array or single sample representing one of the two colour difference signals related to the primary colours

Note 1 to entry: The term chroma is used rather than the term chrominance in order to avoid the implication of the use of linear light transfer characteristics that is often associated with the term chrominance.

3.23

clean random access access unit

CRA access unit

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is a *CRA picture* (3.24)

3.24

clean random access picture

CRA picture

IRAP picture (3.71) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to *CRA_NUT*

Note 1 to entry: A CRA picture does not refer to any pictures other than itself for inter prediction in its decoding process, and can be the first picture in the bitstream in decoding order, or can appear later in the bitstream. A CRA picture can have associated RADL or RASL pictures. As with a BLA picture, a CRA picture can contain syntax elements that specify a non-empty *RPS* (3.131). When a CRA picture has *NoRaslOutputFlag* equal to 1, the associated RASL pictures are not output by the decoder, because they can be non-decodable, as they can contain references to pictures that are not present in the bitstream.

3.25

coded picture

coded representation of a picture containing all *CTUs* (3.34) of the picture

3.26

coded picture buffer

CPB

first-in first-out buffer containing *decoding units* (3.46) in *decoding order* (3.44) specified in the *hypothetical reference decoder* (3.56)

Note 1 to entry: The hypothetical reference decoder is defined in Annex C.

3.27**coded representation**

data element as represented in its coded form

3.28**coded slice segment NAL unit**

NAL unit (3.87) that has *nal_unit_type* in the range of *TRAIL_N* to *RASL_R*, inclusive, or in the range of *BLA_W_LP* to *RSV_IRAP_VCL23*, inclusive, which indicates that the NAL unit contains a coded *slice segment* (3.145)

3.29**coded layer-wise video sequence****CLVS**

sequence of *pictures* (3.99) and the associated *non-VCL NAL units* (3.91) of the base layer of a *CVS* (3.30)

3.30**coded video sequence****CVS**

sequence of *access units* (3.1) that consists, in *decoding order* (3.44), of an *IRAP access unit* (3.65) with *NoRaslOutputFlag* equal to 1, followed by zero or more access units that are not IRAP access units with *NoRaslOutputFlag* equal to 1, including all subsequent access units up to but not including any subsequent access unit that is an IRAP access unit with *NoRaslOutputFlag* equal to 1

Note 1 to entry: An IRAP access unit can be an IDR access unit, a BLA access unit, or a CRA access unit. The value of *NoRaslOutputFlag* is equal to 1 for each IDR access unit, each BLA access unit, and each CRA access unit that is the first access unit in the bitstream in decoding order, is the first access unit that follows an end of sequence NAL unit in decoding order, or has *HandleCraAsBlaFlag* equal to 1.

3.31**coded video sequence group****CVSG**

one or more consecutive *CVSs* (3.30) in *decoding order* (3.44) that collectively consist of an *IRAP access unit* (3.65) that activates a VPS RBSP *firstVpsRbsp* that was not already active followed by all subsequent access units, in decoding order, for which *firstVpsRbsp* is the active VPS RBSP up to the end of the *bitstream* (3.13) or up to but excluding the access unit that activates a different VPS RBSP than *firstVpsRbsp*, whichever is earlier in decoding order

3.32**coding block**

NxN block (3.14) of samples for some value of *N* such that the division of a *CTB* (3.33) into coding blocks is a *partitioning* (3.98)

3.33**coding tree block****CTB**

NxN block (3.14) of samples for some value of *N* such that the division of a *component* (3.36) into CTBs is a *partitioning* (3.98)

3.34**coding tree unit****CTU**

CTB (3.33) of *luma* (3.85) samples, two corresponding CTBs of *chroma* (3.22) samples of a *picture* (3.99) that has three sample arrays, or a CTB of samples of a monochrome picture or a picture that is coded using three separate colour planes and *syntax structures* (3.163) used to code the samples

3.35

coding unit

coding block (3.32) of *luma* (3.85) samples, two corresponding coding blocks of *chroma* (3.22) samples of a *picture* (3.99) that has three sample arrays, or a coding block of samples of a monochrome picture or a picture that is coded using three separate colour planes and *syntax structures* (3.163) used to code the samples

3.36

component

array or single sample from one of the three arrays [*luma* (3.85) and two *chroma* (3.22)] that compose a *picture* (3.99) in 4:2:0, 4:2:2, or 4:4:4 colour format or the array or a single sample of the array that compose a picture in monochrome format

3.37

constituent picture

part of a spatially *frame* (3.57)-packed stereoscopic *picture* (3.99) that corresponds to one view, or a picture itself when frame packing is not in use or the temporal interleaving frame packing arrangement is in use

3.38

context variable

variable specified for the adaptive binary arithmetic *decoding process* (3.45) of a *bin* (3.8) by a formula containing recently decoded bins

3.39

cropped decoded picture

result of cropping a *decoded picture* (3.40) based on the conformance cropping window specified in the *SPS* (3.140) that is referred to by the corresponding *coded picture* (3.25)

3.40

decoded picture

picture derived by decoding a *coded picture* (3.25)

3.41

decoded picture buffer

DPB

buffer holding *decoded pictures* (3.40) for reference, output reordering, or output delay specified for the *hypothetical reference decoder* (3.61)

3.42

decoder

embodiment of a *decoding process* (3.45)

3.43

decoder under test

DUT

decoder (3.42) that is tested for conformance by operating the *hypothetical stream scheduler* (3.62) to deliver a conforming *bitstream* (3.13) to the decoder and to the *hypothetical reference decoder* (3.61) and comparing the values and timing or order of the output of the two decoders

3.44

decoding order

order in which *syntax elements* (3.162) are processed by the *decoding process* (3.45)

3.45**decoding process**

process that reads a *bitstream* (3.13) and derives *decoded pictures* (3.40) from it

Note 1 to entry: The decoding process is specified in this document.

3.46**decoding unit**

access unit (3.1) if SubPicHrdFlag is equal to 0 or a subset of an access unit otherwise, consisting of one or more *VCL NAL units* (3.181) in an access unit and the *associated non-VCL NAL units* (3.4)

3.47**dependent slice segment**

slice segment (3.145) for which the values of some *syntax elements* (3.162) of the *slice segment header* (3.146) are inferred from the values for the preceding *independent slice segment* (3.63) in *decoding order* (3.44)

3.48**display process**

process having, as its input, the *cropped decoded pictures* (3.39) that are the output of the *decoding process* (3.45)

Note 1 to entry: The display process is not specified in this document.

3.49**elementary stream**

sequence of one or more *bitstreams* (3.13)

Note 1 to entry: An elementary stream that consists of two or more bitstreams would typically have been formed by splicing together two or more bitstreams (or parts thereof).

3.50**elevation circle**

circle on a sphere connecting all points with the same elevation value

Note 1 to entry: An elevation circle is similar to a latitude line on the earth. Except when the elevation value is zero, an elevation circle is not a *great circle* (3.60) like a longitude circle on the earth.

3.51**emulation prevention byte**

byte (3.19) equal to 0x03 that is present within a *NAL unit* (3.87) when the *syntax elements* (3.162) of the *bitstream* (3.13) form certain patterns of byte values in a manner that ensures that no sequence of consecutive *byte-aligned* (3.20) bytes in the NAL unit can contain a *start code prefix* (3.150)

3.52**encoder**

embodiment of an *encoding process* (3.53)

3.53**encoding process**

process that produces a *bitstream* (3.13)

Note 1 to entry: The bitstream produced conforms to this document. This process is not specified in this document.

3.54

field

assembly of alternative rows of samples of a *frame* (3.57)

3.55

filler data NAL unit

NAL unit (3.87) with *nal_unit_type* equal to FD_NUT

3.56

flag

variable or single-bit *syntax element* (3.162) that can take one of the two possible values, 0 and 1

3.57

frame

composition of a top *field* (3.54) and a bottom field, where sample rows 0, 2, 4, ... originate from the top field and sample rows 1, 3, 5, ... originate from the bottom field

3.58

frequency index

one-dimensional or two-dimensional index associated with a *transform coefficient* (3.175) prior to the application of a *transform* (3.173) in the *decoding process* (3.45)

3.59

global coordinate axes

coordinate axes associated with *omnidirectional video* (3.92) that are associated with an externally referenceable position and orientation

Note 1 to entry: The global coordinate axes could correspond to the position and orientation of a device or rig used for omnidirectional audio/video acquisition as well as the position of an observer's head in the three-dimensional space of the omnidirectional video rendering environment.

3.60

great circle

intersection of a sphere and a plane that passes through the centre point of the sphere

Note 1 to entry: A great circle is also known as an orthodrome or Riemannian circle.

3.61

hypothetical reference decoder

HRD

hypothetical *decoder* (3.42) model that specifies constraints on the variability of conforming *NAL unit streams* (3.88) or conforming *byte streams* (3.21) that an *encoding process* (3.53) may produce

3.62

hypothetical stream scheduler

HSS

hypothetical delivery mechanism used for checking the conformance of a *bitstream* (3.13) or a *decoder* (3.42) with regards to the timing and data flow of the input of a bitstream into the *hypothetical reference decoder* (3.61)

3.63

independent slice segment

slice segment (3.145) for which the values of the *syntax elements* (3.162) of the *slice segment header* (3.146) are not inferred from the values for a preceding slice segment

3.64**instantaneous decoding refresh access unit****IDR access unit**

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is an *IDR picture* (3.65)

3.65**instantaneous decoding refresh picture****IDR picture**

IRAP picture (3.71) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to *IDR_W_RADL* or *IDR_N_LP*

Note 1 to entry: An IDR picture does not refer to any pictures other than itself for inter prediction in its decoding process, and can be the first picture in the bitstream in decoding order, or can appear later in the bitstream. Each IDR picture is the first picture of a CVS in decoding order. When an IDR picture for which each VCL NAL unit has *nal_unit_type* equal to *IDR_W_RADL*, it can have associated RADL pictures. When an IDR picture for which each VCL NAL unit has *nal_unit_type* equal to *IDR_N_LP*, it does not have any associated leading pictures. An IDR picture does not have associated RASL pictures.

3.66**inter coding**

coding of a *coding block* (3.33), *slice* (3.143), or *picture* (3.99) that uses *inter prediction* (3.67)

3.67**inter prediction**

prediction (3.103) derived in a manner that is dependent on data elements (e.g. sample values or motion vectors) of one or more *reference pictures* (3.128)

Note 1 to entry: A prediction from a reference picture that is the current picture itself is also inter prediction.

3.68**intra coding**

coding of a *coding block* (3.33), *slice* (3.143), or *picture* (3.99) that uses *intra prediction* (3.69)

3.69**intra prediction**

prediction (3.103) derived from only data elements (e.g. sample values) of the same decoded *slice* (3.143) without referring to a *reference picture* (3.128)

3.70**intra random access point access unit****IRAP access unit**

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is an *IRAP picture* (3.71)

3.71**intra random access point picture****IRAP picture**

coded picture (3.25) for which each *VCL NAL unit* (3.181) has *nal_unit_type* in the range of *BLA_W_LP* to *RSV_IRAP_VCL23*, inclusive

Note 1 to entry: An IRAP picture does not refer to any pictures other than itself for inter prediction in its decoding process, and can be a BLA picture, a CRA picture or an IDR picture. The first picture in the bitstream in decoding order is an IRAP picture. Provided the necessary parameter sets are available when they are activated, the IRAP picture and all subsequent non-RASL pictures in decoding order can be correctly decoded without performing the decoding process of any pictures that precede the IRAP picture in decoding order. There can be pictures in a bitstream that do not refer to any pictures other than itself for inter prediction in its decoding process that are not IRAP pictures.

3.72

intra slice

I slice

slice (3.143) that is decoded using *intra prediction* (3.69) only

3.73

layer

<scalable> set of *VCL NAL units* (3.181) that all have a particular value of *nuh_layer_id* and the associated *non-VCL NAL units* (3.4)

Note 1 to entry: Depending on the context, either the first layer concept or the second layer concept applies. This first layer concept is also referred to as a scalable layer, wherein a layer can be a spatial scalable layer, a quality scalable layer, a view, etc. A temporal true subset of a scalable layer is not referred to as a layer but referred to as a sub-layer or temporal sub-layer.

3.74

layer identifier list

list of *nuh_layer_id* values that is associated with a *layer set* (3.75) or an *operation point* (3.93) and can be used as an input to the *sub-bitstream extraction process* (3.154)

3.75

layer set

set of layers represented within a *bitstream* (3.13) created from another bitstream by operation of the *sub-bitstream extraction process* (3.154) with the another bitstream, the target highest TemporalId equal to 6, and the target *layer identifier list* (3.74) equal to the layer identifier list associated with the layer set as inputs

3.76

leading picture

picture (3.99) that precedes the associated *IRAP picture* (3.71) in *output order* (3.94)

3.77

leaf

terminating node of a *tree* (3.178) that is a root node of a tree of depth 0

3.78

level

<constraint> defined set of constraints on the values that may be taken by the *syntax elements* (3.162) and variables of this document

Note 1 to entry: 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 could, within the specified constraints, support a different level for each supported profile.

3.79

level

<transform coefficient> value of a *transform coefficient* (3.175) prior to *scaling* (3.139)

3.80

list 0 (list 1) motion vector

motion vector (3.86) associated with a reference index pointing into reference picture list 0 (list 1)

3.81

list 0 (list 1) prediction

inter prediction (3.67) of the content of a *slice* (3.143) using a reference index pointing into reference picture list 0 (list 1)

3.82**local coordinate axes**

coordinate axes having a specified rotation relationship relative to the *global coordinate axes* (3.59)

3.83**long-term reference picture**

picture (3.99) that is marked as “used for long-term reference”

3.84**long-term reference picture set**

two *RPS* (3.132) lists that may contain *long-term reference pictures* (3.83)

3.85**luma**

Y or L

sample array or single sample representing the monochrome signal related to the primary colours

Note 1 to entry: 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.

3.86**motion vector**

two-dimensional vector used for *inter prediction* (3.67) that provides an offset from the coordinates in the *decoded picture* (3.40) to the coordinates in a *reference picture* (3.128)

3.87**network abstraction layer unit****NAL unit**

syntax structure (3.163) containing an indication of the type of data to follow and *bytes* (3.19) containing that data in the form of an *RBSP* (3.124) interspersed as necessary with *emulation prevention bytes* (3.51)

3.88**network abstraction layer unit stream****NAL unit stream**

sequence of *NAL units* (3.87)

3.89**non-scalable-nested SEI message**

SEI message that is not contained in a scalable nesting SEI message

3.90**non-reference picture**

picture (3.99) that is marked as “unused for reference”

Note 1 to entry: A non-reference picture contains samples that cannot be used for inter prediction in the decoding process of subsequent pictures in decoding order. In other words, once a picture is marked as “unused for reference”, it can never be marked back as “used for reference”.

3.91**non-VCL NAL unit**

NAL unit (3.87) that is not a *VCL NAL unit* (3.181)

3.92**omnidirectional video**

video content in a format that enables rendering according to the user's viewing orientation, e.g., if viewed using a head-mounted device, or according to a user's desired *viewport* (3.183), reflecting a potentially rotated viewing position

3.93

operation point

bitstream (3.13) created from another bitstream by operation of the *sub-bitstream extraction process* (3.154) with another bitstream, a target highest TemporalId, and a target *layer identifier list* (3.74) as inputs

Note 1 to entry: If the target highest TemporalId of an operation point is equal to the greatest value of TemporalId in the layer set associated with the target layer identification list, the operation point is identical to the layer set. Otherwise it is a subset of the layer set.

3.94

output order

order in which the *decoded pictures* (3.40) are output from the *decoded picture buffer* (3.41) for the decoded pictures that are to be output from the decoded picture buffer

3.95

output time

time when a *decoded picture* (3.40) is to be output as specified by the *HRD* (3.56) according to the output timing *decoded picture buffer* (3.41) operation

3.96

packed region

region in a *region-wise packed picture* (3.133) that is mapped to a *projected region* (3.113) according to a *region-wise packing* (3.134)

3.97

parameter

syntax element of a *VPS* (3.182), *SPS* (3.140) or *PPS* (3.100), or the second word of the defined term *quantization parameter* (3.117)

3.98

partitioning

division of a set into subsets such that each element of the set is in exactly one of the subsets

3.99

picture

array of *luma* (3.85) samples in monochrome format or an array of luma samples and two corresponding arrays of *chroma* (3.22) samples in 4:2:0, 4:2:2, and 4:4:4 colour format

Note 1 to entry: A picture can be either a frame or a field. However, in one CVS, either all pictures are frames or all pictures are fields.

3.100

picture parameter set

PPS

syntax structure (3.163) containing *syntax elements* (3.162) that apply to zero or more entire *coded pictures* (3.25) as determined by a syntax element found in each *slice segment header* (3.146)

3.101

picture order count

POC

variable that is associated with each *picture* (3.99), uniquely identifies the associated picture among all pictures in the *CVS* (3.30), and, when the associated picture is to be output from the *decoded picture buffer* (3.41), indicates the position of the associated picture in *output order* (3.94) relative to the output order positions of the other pictures in the same CVS that are to be output from the decoded picture buffer

3.102

picture unit

set of *NAL units* (3.87) that contain all *VCL NAL units* (3.181) of a *coded picture* (3.25) and their *associated non-VCL NAL units* (3.4)

3.103

prediction

embodiment of the *prediction process* (3.105)

3.104

prediction block

rectangular *MxN block* (3.14) of samples on which the same *prediction* (3.103) is applied

3.105

prediction process

use of a *predictor* (3.108) to provide an estimate of the data element (e.g. sample value or motion vector) currently being decoded

3.106

prediction unit

prediction block (3.104) of *luma* (3.85) samples, two corresponding prediction blocks of *chroma* (3.22) samples of a *picture* (3.99) that has three sample arrays, or a prediction block of samples of a monochrome picture or a picture that is coded using three separate colour planes and *syntax structures* (3.163) used to predict the prediction block samples

3.107

predictive slice

P slice

slice (3.143) that is decoded using *intra prediction* (3.69) or using *inter prediction* (3.67) with at most one *motion vector* (3.86) and *reference index* (3.127) to predict the sample values of each *block* (3.14)

3.108

predictor

combination of specified values or previously decoded data elements (e.g. sample value or motion vector) used in the *decoding process* (3.45) of subsequent data elements

3.109

prefix SEI message

SEI message that is contained in a *prefix SEI NAL unit* (3.110)

3.110

prefix SEI NAL unit

SEI NAL unit (3.161) that has *nal_unit_type* equal to PREFIX_SEI_NUT

3.111

profile

specified subset of syntax

Note 1 to entry: The syntax of this document.

3.112

projected picture

picture that uses a *projection* (3.114) format for *omnidirectional video* (3.92)

3.113

projected region

region in a *projected picture* (3.112) that is mapped to a *packed region* (3.96) according to a *region-wise packing* (3.134)

3.114

projection

specified correspondence between the colour samples of a *projected picture* (3.112) and azimuth and elevation positions on a sphere

3.115

pulse code modulation

PCM

coding of the samples of a *block* (3.14) by directly representing the sample values without *prediction* (3.103) or application of a transform

3.116

quadtree

tree (3.178) in which a parent node can be split into four child nodes, each of which may become parent node for another split into four child nodes

3.117

quantization parameter

variable used by the *decoding process* (3.45) for scaling of transform coefficient levels

3.118

random access

act of starting the decoding process for a *bitstream* (3.13) at a point other than the beginning of the stream

3.119

random access decodable leading access unit

RADL access unit

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is a *RADL picture* (3.120)

3.120

random access decodable leading picture

RADL picture

coded picture for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to RADL_R or RADL_N

Note 1 to entry: All RADL pictures are leading pictures. RADL pictures are not used as reference pictures for the decoding process of trailing pictures of the same associated IRAP picture. When present, all RADL pictures precede, in decoding order, all trailing pictures of the same associated IRAP picture.

3.121

random access skipped leading access unit

RASL access unit

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is a RASL picture

3.122

random access skipped leading picture

RASL picture

coded picture (3.25) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to RASL_R or RASL_N

Note 1 to entry: All RASL pictures are leading pictures of an associated BLA or CRA picture. When the associated IRAP picture has *NoRaslOutputFlag* equal to 1, the RASL picture is not output and can be non-correctly-decodable, as the RASL picture can contain references to pictures that are not present in the bitstream. RASL pictures are not used as reference pictures for the decoding process of non-RASL pictures. When present, all RASL pictures precede, in decoding order, all trailing pictures of the same associated IRAP picture.

3.123**raster scan**

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

3.124**raw byte sequence payload****RBSP**

syntax structure (3.163) containing an integer number of *bytes* (3.19) that is encapsulated in a *NAL unit* (3.87) and that is either empty or has the form of a *string of data bits* (3.153) containing *syntax elements* (3.162) followed by an *RBSP stop bit* (3.125) and zero or more subsequent bits equal to 0

3.125**raw byte sequence payload stop bit****RBSP stop bit**

bit equal to 1 present within a *raw byte sequence payload* (3.124) after a *string of data bits* (3.153), for which the location of the end 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

3.126**recovery point**

point in the *bitstream* (3.13) at which the recovery of an exact or an approximate representation of the *decoded pictures* (3.40) represented by the bitstream is achieved after a *random access* (3.118) or *broken link* (3.15)

3.127**reference index**

index into a *reference picture list* (3.129)

3.128**reference picture**

picture (3.99) that is a *short-term reference picture* (3.141) or a *long-term reference picture* (3.83)

Note 1 to entry: A reference picture contains samples that can be used for inter prediction in the decoding process of subsequent pictures in decoding order.

3.129**reference picture list**

list of *reference pictures* (3.128) that is used for *inter prediction* (3.67) of a *P slice* (3.107) or *B slice* (3.12)

Note 1 to entry: For the decoding process of a P slice, there is one reference picture list – reference picture list 0. For the decoding process of a B slice, there are two reference picture lists – reference picture list 0 and reference picture list 1.

3.130

reference picture list 0

reference picture list (3.129) used for *inter prediction* (3.67) of a *P slice* (3.107) or the first reference picture list used for *inter prediction* of a *B slice* (3.12)

3.131

reference picture list 1

second *reference picture list* (3.129) used for *inter prediction* (3.67) of a *B slice* (3.12)

3.132

reference picture set

RPS

set of *reference pictures* (3.128) associated with a *picture* (3.99), consisting of all reference pictures that are prior to the associated picture in *decoding order* (3.44), that may be used for *inter prediction* (3.67) of the associated picture or any picture following the associated picture in decoding order

Note 1 to entry: The RPS of a picture consists of five RPS lists, three of which are to contain short-term reference pictures and the other two are to contain long-term reference pictures.

3.133

region-wise packed picture

decoded picture (3.40) that contains one or more *packed regions* (3.96)

Note 1 to entry: A region-wise packed picture can contain a *region-wise packing* (3.134) of a *projected picture* (3.112).

3.134

region-wise packing

transformation, resizing, and relocation of *packed regions* (3.96) of a *region-wise packed picture* (3.133) to remap the packed regions to *projected regions* (3.113) of a *projected picture* (3.112)

3.135

reserved

<value of a particular syntax element> value that may be used in the future by ITU-T | ISO/IEC

Note 1 entry: These values cannot be used in *bitstreams* (3.13) conforming to this document, but can be used in *bitstreams* conforming to future editions of this document by ITU-T | ISO/IEC.

3.136

residual

decoded difference between a *prediction* (3.103) of a sample or data element and its decoded value

3.137

sample aspect ratio

ratio between the intended horizontal distance between the columns and the intended vertical distance between the rows of the *luma* (3.85) sample array in a *picture* (3.99), which is specified for assisting the *display process* (3.48) (not specified in this document) and expressed as *h:v*, where *h* is the horizontal width and *v* is the vertical height, in arbitrary units of spatial distance

3.138

scalable-nested SEI message

SEI message that is contained in a scalable nesting SEI message

3.139

scaling

process of multiplying *transform coefficient levels* (3.176) by a factor, resulting in *transform coefficients* (3.175)

3.140

sequence parameter set

SPS

syntax structure (3.163) containing *syntax elements* (3.162) that apply to zero or more entire *CVSs* (3.30) as determined by the content of a syntax element found in the *PPS* (3.100) referred to by a syntax element found in each *slice segment header* (3.146)

3.141

short-term reference picture

picture (3.99) that is marked as “used for short-term reference”

3.142

short-term reference picture set

three *RPS* (3.132) lists that may contain short-term reference pictures

3.143

slice

integer number of *CTUs* (3.34) contained in one *independent slice segment* (3.58) and all subsequent *dependent slice segments* (3.47) (if any) that precede the next independent slice segment (if any) within the same *access unit* (3.1)

3.144

slice header

slice segment header (3.146) of the *independent slice segment* (3.63) that is a current *slice segment* (3.145) or the most recent independent slice segment that precedes a current *dependent slice segment* (3.47) in *decoding order* (3.44)

3.145

slice segment

integer number of *CTUs* (3.34) ordered consecutively in the *tile scan* (3.170) and contained in a single *NAL unit* (3.87)

3.146

slice segment header

part of a coded *slice segment* (3.145) containing the data elements pertaining to the first or all *CTUs* (3.34) represented in the slice segment

3.147

source

video material or some of its attributes before encoding

3.148

sphere coordinates

azimuth and elevation angles identifying a location of a point on a sphere

3.149

sphere region

region on a sphere, specified either by four *great circles* (3.60) or by two *azimuth circles* (3.6) and two *elevation circles* (3.50), or such a region on a rotated sphere after applying yaw, pitch, and roll rotations

3.150

start code prefix

unique sequence of three *bytes* (3.19) equal to 0x000001 embedded in the *byte stream* (3.21) as a prefix to each *NAL unit* (3.87)

Note 1 to entry: 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.

3.151

step-wise temporal sub-layer access access unit

STSA access unit

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is an *STSA picture* (3.152)

3.152

step-wise temporal sub-layer access picture

STSA picture

coded picture (3.25) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to STSA_R or STSA_N

Note 1 to entry: An STSA picture does not use pictures with the same TemporalId as the STSA picture for inter prediction reference. Pictures following an STSA picture in decoding order with the same TemporalId as the STSA picture do not use pictures prior to the STSA picture in decoding order with the same TemporalId as the STSA picture for inter prediction reference. An STSA picture enables up-switching, at the STSA picture, to the sub-layer containing the STSA picture, from the immediately lower sub-layer. STSA pictures have TemporalId greater than 0.

3.153

string of data bits

SODB

sequence of some number of bits representing *syntax elements* (3.162) present within a *raw byte sequence payload* (3.124) prior to the *raw byte sequence payload stop bit* (3.125), where 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

3.154

sub-bitstream extraction process

specified process by which *NAL units* (3.87) in a *bitstream* (3.13) that do not belong to a target set, determined by a target highest TemporalId and a target *layer identifier list* (3.74), are removed from the bitstream, with the output sub-bitstream consisting of the NAL units in the bitstream that belong to the target set

3.155

sub-layer

temporal sub-layer

temporal scalable layer of a temporal scalable *bitstream* (3.13), consisting of *VCL NAL units* (3.181) with a particular value of the TemporalId variable and the associated *non-VCL NAL units* (3.91)

3.156

sub-layer non-reference picture

SLNR picture

picture (3.99) that contains samples that cannot be used for *inter prediction* (3.67) in the *decoding process* (3.43) of subsequent pictures of the same *sub-layer* (3.142) in *decoding order* (3.42)

Note 1 to entry: Samples of an SLNR picture can be used for inter prediction in the decoding process of subsequent pictures of higher sub-layers in decoding order.

3.157

sub-layer reference picture

picture (3.93) that contains samples that may be used for *inter prediction* (3.62) in the *decoding process* (3.45) of subsequent pictures of the same *sub-layer* (3.155) in *decoding order* (3.44)

Note 1 to entry: Samples of a sub-layer reference picture can also be used for inter prediction in the decoding process of subsequent pictures of higher sub-layers in decoding order.

3.158**sub-layer representation**

subset of the *bitstream* (3.13) consisting of *NAL units* (3.87) of a particular *sub-layer* (3.155) and the lower sub-layers

3.159**suffix SEI message**

SEI message that is contained in a *suffix SEI NAL unit* (3.160)

3.160**suffix SEI NAL unit**

SEI NAL unit (3.161) that has *nal_unit_type* equal to SUFFIX_SEI_NUT

3.161**supplemental enhancement information NAL unit****SEI NAL unit**

NAL unit (3.87) that has *nal_unit_type* equal to PREFIX_SEI_NUT or SUFFIX_SEI_NUT

3.162**syntax element**

element of data represented in the *bitstream* (3.13)

3.163**syntax structure**

zero or more *syntax elements* (3.162) present together in the *bitstream* (3.13) in a specified order

3.164**temporal sub-layer access access unit****TSA access unit**

access unit (3.1) in which the *coded picture* (3.25) with *nuh_layer_id* equal to 0 is a *TSA picture* (3.165)

3.165**temporal sub-layer access picture****TSA picture**

coded picture (3.25) for which each *VCL NAL unit* (3.181) has *nal_unit_type* equal to TSA_R or TSA_N

Note 1 to entry: A TSA picture and pictures following the TSA picture in decoding order do not use pictures prior to the TSA picture in decoding order with *TemporalId* greater than or equal to that of the TSA picture for inter prediction reference. A TSA picture enables up-switching, at the TSA picture, to the sub-layer containing the TSA picture or any higher sub-layer, from the immediately lower sub-layer. TSA pictures have *TemporalId* greater than 0.

3.166**tier**

specified category of *level* (3.79) constraints imposed on values of the *syntax elements* (3.162) in the *bitstream* (3.13)

Note 1 to entry: The level constraints are nested within a tier and a *decoder* (3.42) conforming to a certain tier and level would be capable of decoding all bitstreams that conform to the same tier or the lower tier of that level or any level below it.

3.167

tile

rectangular region of *CTUs* (3.34) within a particular *tile column* (3.168) and a particular *tile row* (3.169) in a *picture* (3.99)

3.168

tile column

rectangular region of *CTUs* (3.34) having a height equal to the height of the *picture* (3.99) and a width specified by *syntax elements* (3.162) in the *picture parameter set* (3.100)

3.169

tile row

rectangular region of *CTUs* (3.34) having a height specified by *syntax elements* (3.162) in the *picture parameter set* (3.100) and a width equal to the width of the *picture* (3.99)

3.170

tile scan

specific sequential ordering of *CTUs* (3.34) *partitioning* (3.98) a *picture* (3.99) in which the *CTUs* are ordered consecutively in *CTUs raster scan* (3.123) in a *tile* (3.167)

Note 1 to entry: Tiles in a picture are ordered consecutively in a raster scan of the tiles of the picture.

3.171

tilt angle

angle indicating the amount of tilt of a *sphere region* (3.149), measured as the amount of rotation of a sphere region along the axis originating from the sphere origin passing through the centre point of the sphere region, where the angle value increases clockwise when looking from the origin towards the positive end of the axis

3.172

trailing picture

non-IRAP *picture* (3.99) that follows the *associated IRAP picture* (3.3) in *output order* (3.94)

Note 1 to entry: Trailing pictures associated with an IRAP picture also follow the IRAP picture in decoding order. Pictures that follow the associated IRAP picture in output order and precede the associated IRAP picture in decoding order are not allowed.

3.173

transform

part of the *decoding process* (3.45) by which a *block* (3.14) of *transform coefficients* (3.175) is converted to a block of spatial-domain values

3.174

transform block

rectangular *MxN block* (3.14) of samples resulting from a *transform* (3.173) in the *decoding process* (3.45)

3.175

transform coefficient

scalar quantity, considered to be in a frequency domain, that is associated with a particular one-dimensional or two-dimensional *frequency index* (3.58) in a *transform* (3.173) in the *decoding process* (3.45)

3.176**transform coefficient level**

integer quantity representing the value associated with a particular two-dimensional frequency index in the *decoding process* (3.45) prior to scaling for computation of a transform coefficient value

3.177**transform unit**

transform block (3.174) of *luma* (3.85) samples of size 8×8, 16×16, or 32×32 or four transform blocks of luma samples of size 4×4, two corresponding transform blocks of *chroma* (3.22) samples of a *picture* (3.99) in 4:2:0 colour format; or transform block of luma samples of size 8×8, 16×16, or 32×32, and four corresponding transform blocks of chroma samples, or four transform blocks of luma samples of size 4×4, and four corresponding transform blocks of chroma samples of a picture in 4:2:2 colour format; or transform block of luma samples of size 4×4, 8×8, 16×16, or 32×32, and two corresponding transform blocks of chroma samples of a picture in 4:4:4 colour format that is not coded using three separate colour planes and *syntax structures* (3.163) used to transform the transform block samples; or transform block of luma samples of size 8×8, 16×16, or 32×32 or four transform blocks of luma samples of size 4×4 of a monochrome picture or a picture in 4:4:4 colour format that is coded using three separate colour planes; and the associated syntax structures used to transform the transform block samples

3.178**tree**

finite set of nodes with a unique root node

3.179**universal unique identifier****UUID**

identifier that is unique with respect to the space of all universal unique identifiers

3.180**unspecified**

<value of a particular syntax element> value not to be used in the future by ITU-T/ISO/IEC

Note 1 to entry: These values have no specified meaning in this document and will not have a specified meaning in the future as an integral part of future editions of this document.

3.181**video coding layer NAL unit****VCL NAL unit**

collective term for *coded slice segment NAL units* (3.28) and the subset of *NAL units* (3.87) that have *reserved* (3.135) values of *nal_unit_type* that are classified as VCL NAL units

3.182**video parameter set****VPS**

syntax structure (3.163) containing *syntax elements* (3.162) that apply to zero or more entire *CVSs* (3.30) as determined by the content of a syntax element found in the *SPS* (3.140) referred to by a syntax element found in the *PPS* (3.100) referred to by a syntax element found in each *slice segment header* (3.146)

3.183**viewport**

region of *omnidirectional video* (3.92) content suitable for display and viewing by the user

3.184**z-scan order**

specified sequential ordering of *blocks* (3.14) *partitioning* (3.98) a *picture* (3.99) where the order is identical to *CTBs* (3.33) *raster scan* (3.123) of the picture when the blocks are of the same size as CTBs and, when the blocks are of a smaller size than CTBs, i.e. CTBs are further partitioned into smaller *coding blocks* (3.32), the order traverses from CTB to CTB in CTB raster scan of the picture, and inside each CTB, which may be divided into *quadtrees* (3.116) hierarchically to lower levels, the order traverses from quadtree to quadtree of a particular level in quadtree-of-the-particular-level raster scan of the quadtree of the immediately higher level

4 Abbreviated terms

ATSC	Advanced Television Systems Committee
B	bi-predictive
BLA	broken link access
BPB	bitstream partition buffer
CABAC	context-based adaptive binary arithmetic coding
CB	coding block
CBR	constant bit rate
CLVS	coded layer-wise video sequence
CPB	coded picture buffer
CRA	clean random access
CTB	coding tree block
CTU	coding tree unit
CU	coding unit
CVS	coded video sequence
CVSG	coded video sequence group
DCT	discrete cosine transform
DPB	decoded picture buffer
DRAP	dependent random access point
DUT	decoder under test
EG	exponential-Golomb
EGk	k-th order exponential-Golomb
FCC	federal communications commission (of the United States)
FIFO	first-in, first-out
FIR	finite impulse response
FL	fixed-length
GBR	green, blue, and red
GDR	gradual decoding refresh
HRD	hypothetical reference decoder
HSS	hypothetical stream scheduler

I	intra
IDCT	inverse discrete cosine transformation
IDR	instantaneous decoding refresh
INBLD	independent non-base layer decoding
IRAP	intra random access point
LPS	least probable symbol
LSB	least significant bit
MCTS	motion-constrained tile set
MPS	most probable symbol
MSB	most significant bit
MVP	motion vector prediction
NAL	network abstraction layer
NTSC	national television system committee (of the United States)
OLS	output layer set
P	predictive
PB	prediction block
PCM	pulse code modulation
POC	picture order count
PPS	picture parameter set
PU	prediction unit
QP	quantization parameter
RADL	random access decodable leading (picture)
RASL	random access skipped leading (picture)
RBSP	raw byte sequence payload
RGB	red, green, and blue
RPS	reference picture set
SAO	sample adaptive offset
SAR	sample aspect ratio
SEI	supplemental enhancement information
SLNR	sub-layer non-reference (picture)
SMPTE	society of motion picture and television engineers
SODB	string of data bits
SPS	sequence parameter set
STSA	step-wise temporal sub-layer access
TB	transform block
TR	truncated Rice
TSA	temporal sub-layer access

TU	transform unit
UCS	universal coded character set
UTF	universal coded character set transmission format
UUID	universal unique identifier
VBR	variable bit rate
VCL	video coding layer
VPS	video parameter set
VUI	video usability information

5 Conventions

5.1 General

When the word “shall” is 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 “shall” is used in reference to operations performed by the decoding process, any decoding process that produces identical cropped decoded pictures to those output from the decoding process described in this document conforms to the decoding process requirements of this document.

NOTE The mathematical operators used in this document are similar to those used in the C programming language. However, the results of integer division and arithmetic shift operations are defined more precisely, and additional operations are defined, such as exponentiation and real-valued division. Numbering and counting conventions generally begin from 0, e.g. “the first” is equivalent to the 0-th, “the second” is equivalent to the 1-th, etc.

5.2 Arithmetic operators

+	addition
–	subtraction (as a two-argument operator) or negation (as a unary prefix operator)
*	multiplication, including matrix 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.
÷	division in mathematical formulae where no truncation or rounding is intended
$\frac{x}{y}$	division in mathematical formulae where no truncation or rounding is intended
$\sum_{i=x}^y f(i)$	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 $x \geq 0$ and $y > 0$.

5.3 Logical operators

$x \&\& y$	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

5.4 Relational operators

>	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.

5.5 Bit-wise operators

&	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.
^	bit-wise “exclusive 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.
$x \gg y$	arithmetic right shift of a two's complement integer representation of x by y binary digits	This function is defined only for non-negative integer values of y . Bits shifted into the MSBs as a result of the right shift have a value equal to the MSB of x prior to the shift operation.
$x \ll y$	arithmetic left shift of a two's complement integer representation of x by y binary digits	This function is defined only for non-negative integer values of y . Bits shifted into the LSBs as a result of the left shift have a value equal to 0.

5.6 Assignment operators

=	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 $x = x - (-3)$

5.7 Range notation

$x = y..z$ x takes on integer values starting from y to z , inclusive, with x , y , and z being integer numbers and z being greater than or equal to y

5.8 Mathematical functions

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

$\text{Asin}(x)$ the trigonometric inverse sine function, operating on an argument x that is in the range of -1.0 to 1.0 , inclusive, with an output value in the range of $-\pi/2$ to $\pi/2$, inclusive, in units of radians (5-2)

$\text{Atan}(x)$ the trigonometric inverse tangent function, operating on an argument x , with an output value in the range of $-\pi/2$ to $\pi/2$, inclusive, in units of radians (5-3)

$$\text{Atan2}(y, x) = \begin{cases} \text{Atan}\left(\frac{y}{x}\right) & ; \text{ if } x > 0 \\ \text{Atan}\left(\frac{y}{x}\right) + \pi & ; \text{ if } x < 0 \ \&\& \ y \geq 0 \\ \text{Atan}\left(\frac{y}{x}\right) - \pi & ; \text{ if } x < 0 \ \&\& \ y < 0 \\ +\frac{\pi}{2} & ; \text{ if } x == 0 \ \&\& \ y \geq 0 \\ -\frac{\pi}{2} & ; \text{ otherwise} \end{cases} \quad (5-4)$$

$\text{Ceil}(x)$ the smallest integer greater than or equal to x (5-5)

$\text{Clip1}_Y(x) = \text{Clip3}(0, (1 \ll \text{BitDepth}_Y) - 1, x)$ (5-6)

$\text{Clip1}_C(x) = \text{Clip3}(0, (1 \ll \text{BitDepth}_C) - 1, x)$ (5-7)

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; z < x \\ y & ; z > y \\ z & ; \text{ otherwise} \end{cases} \quad (5-8)$$

$\text{Cos}(x)$ the trigonometric cosine function operating on an argument x in units of radians (5-9)

Floor(x) the largest integer less than or equal to x (5-10)

$$\text{GetCurrMsb}(a, b, c, d) = \begin{cases} c + d & ; \quad b - a \geq d / 2 \\ c - d & ; \quad a - b > d / 2 \\ c & ; \quad \text{otherwise} \end{cases} \quad (5-11)$$

Ln(x) the natural logarithm of x (the base-e logarithm, where e is the natural logarithm base constant 2.718 281 828...) (5-12)

Log2(x) the base-2 logarithm of x (5-13)

Log10(x) the base-10 logarithm of x (5-14)

$$\text{Min}(x, y) = \begin{cases} x & ; \quad x \leq y \\ y & ; \quad x > y \end{cases} \quad (5-15)$$

$$\text{Max}(x, y) = \begin{cases} x & ; \quad x \geq y \\ y & ; \quad x < y \end{cases} \quad (5-16)$$

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5) \quad (5-17)$$

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x > 0 \\ 0 & ; \quad x = 0 \\ -1 & ; \quad x < 0 \end{cases} \quad (5-18)$$

Sin(x) the trigonometric sine function operating on an argument x in units of radians (5-19)

Sqrt(x) the square root of x (5-20)

Swap(x, y) = (y, x) (5-21)

Tan(x) the trigonometric tangent function operating on an argument x in units of radians (5-22)

5.9 Order of operation precedence

When order of precedence in an expression is not indicated explicitly by use of parentheses, the following rules apply:

- Operations of a higher precedence are evaluated before any operation of a lower precedence.
- Operations of the same precedence are evaluated sequentially from left to right.

Table 5-1 specifies the precedence of operations from highest to lowest; a higher position in the table indicates a higher precedence.

NOTE For those operators that are also used in the C programming language, the order of precedence used in this document is the same as used in the C programming language.

Table 5-1 — Operation precedence from highest (at top of table) to lowest (at bottom of table)

Operations (with operands x, y, and z)
"x++", "x--"
"!x", "-x" (as a unary prefix operator)
"x ^y "
"x * y", "x / y", "x ÷ y", " $\frac{x}{y}$ ", "x % y"
"x + y", "x - y" (as a two-argument operator), " $\sum_{i=x}^y f(i)$ "
"x << y", "x >> y"
"x < y", "x <= y", "x > y", "x >= y"
"x = y", "x != y"
"x & y"
"x y"
"x && y"
"x y"
"x ? y : z"
"x.y"
"x = y", "x += y", "x -= y"

5.10 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), and one descriptor 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 letters 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 without mentioning the originating syntax structure of the variable. Variables starting with a lower case letter are only used within the clause in which they are derived.

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 that specify properties of the current position in the bitstream are referred to as syntax functions. These functions are specified in 7.2 and assume the existence of a bitstream pointer with an

indication of the position of the next bit to be read by the decoding process from the bitstream. Syntax functions are described by their names, which are constructed as syntax element names and end 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).

Functions that are not syntax functions (including mathematical functions specified in 5.8) are described by their names, which start with an upper case letter, contain a mixture of lower and upper case letters without any underscore character, and end with left and right 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} . A single column of a matrix may be referred to as a list and denoted by omission of the row index. Thus, the column of a matrix s at horizontal position x may be referred to as the list $s[x]$.

A specification of values of the entries in rows and columns of an array may be denoted by $\{ \{ \dots \} \{ \dots \} \}$, where each inner pair of brackets specifies the values of the elements within a row in increasing column order and the rows are ordered in increasing row order. Thus, setting a matrix s equal to $\{ \{ 1 \ 6 \} \{ 4 \ 9 \} \}$ specifies that $s[0][0]$ is set equal to 1, $s[1][0]$ is set equal to 6, $s[0][1]$ is set equal to 4, and $s[1][1]$ is set equal to 9.

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 (counted from the most to the least significant bit) 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 (counted from the most to the least significant bit) 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 value different from zero.

5.11 Text description of logical operations

In the text, a statement of logical operations as would be described mathematically in the following form:

```
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 mathematically in the following form:

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

may be described in the following manner:

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

- If all of the following conditions are true, statement 0:
 - condition 0a
 - condition 0b
- Otherwise, if one or more 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 mathematically in the following form:

```
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

5.12 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 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.

When invoking a process, the assignment of variables is specified as follows:

- If the variables at the invoking and the process specification do not have the same name, the variables are explicitly assigned to lower case input or output variables of the process specification.
- Otherwise (the variables at the invoking and the process specification have the same name), assignment is implied.

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

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

6.1 Bitstream formats

This clause 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.

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 the use of the byte stream format are outside the scope of this document. The byte stream format is specified in Annex B.

6.2 Source, decoded, and output picture formats

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

The video source that is represented by the bitstream is a sequence of pictures in decoding order.

The source and decoded pictures are each comprised of one or more sample arrays:

- Luma (Y) only (monochrome);
- Luma and two chroma (YCbCr or YCgCo);
- Green, blue and red (GBR, also known as RGB);
- Arrays representing other unspecified monochrome or tri-stimulus colour samplings (for example, YZX, also known as XYZ);

For convenience of notation and terminology in this document, 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 variables SubWidthC, and SubHeightC are specified in Table 6-1, depending on the chroma format sampling structure, which is specified through chroma_format_idc and separate_colour_plane_flag.

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 and separate_colour_plane_flag

chroma_format_idc	separate_colour_plane_flag	Chroma format	SubWidthC	SubHeightC
0	0	monochrome	1	1
1	0	4:2:0	2	2
2	0	4:2:2	2	1
3	0	4:4:4	1	1
3	1	4:4:4	1	1

In monochrome sampling, there is only one sample array, which is nominally 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, depending on the value of separate_colour_plane_flag, the following applies:

- If separate_colour_plane_flag is equal to 0, each of the two chroma arrays has the same height and width as the luma array.
- Otherwise (separate_colour_plane_flag is equal to 1), the three colour planes are separately processed as monochrome sampled pictures.

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 16, inclusive, 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 pictures are shown in Figure 6-1. Alternative chroma sample relative locations may be indicated in video usability information (see Annex E).

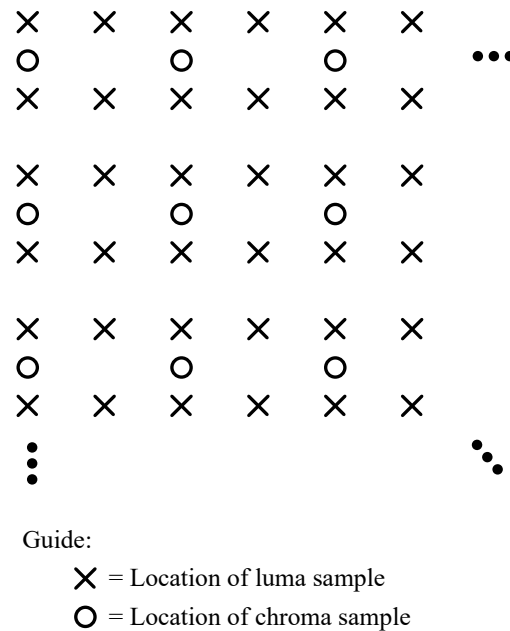


Figure 6-1 — Nominal vertical and horizontal locations of 4:2:0 luma and chroma samples in a picture

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 picture are as shown in Figure 6-2.

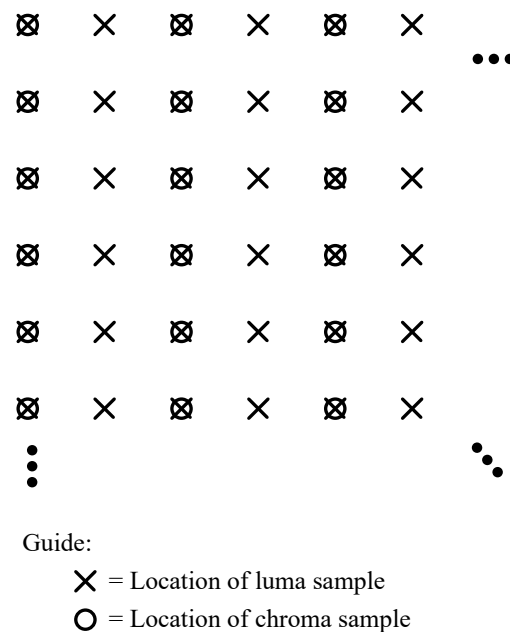


Figure 6-2 — Nominal vertical and horizontal locations of 4:2:2 luma and chroma samples in a picture

When the value of `chroma_format_idc` is equal to 3, all array samples are co-sited for all cases of pictures and the nominal locations in a picture are as shown in Figure 6-3.

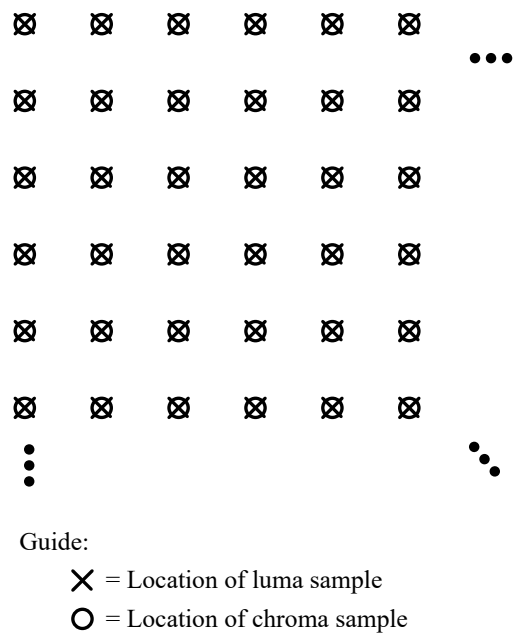


Figure 6-3 — Nominal vertical and horizontal locations of 4:4:4 luma and chroma samples in a picture

6.3 Partitioning of pictures, slices, slice segments, tiles, CTUs, and CTBs

6.3.1 Partitioning of pictures into slices, slice segments, and tiles

This subclause specifies how a picture is partitioned into slices, slice segments, and tiles. Pictures are divided into slices and tiles. A slice is a sequence of one or more slice segments starting with an independent slice segment and containing all subsequent dependent slice segments (if any) that precede the next independent slice segment (if any) within the same picture. A slice segment is a sequence of CTUs. Likewise, a tile is a sequence of CTUs.

For example, a picture may be divided into two slices as shown in Figure 6-4. In this example, the first slice is composed of an independent slice segment containing 4 CTUs, a dependent slice segment containing 32 CTUs, and another dependent slice segment containing 24 CTUs; and the second slice consists of a single independent slice segment containing the remaining 39 CTUs of the picture.

As another example, a picture may be divided into two tiles separated by a vertical tile boundary as shown in Figure 6-5. The left side of the figure illustrates a case in which the picture only contains one slice, starting with an independent slice segment and followed by four dependent slice segments. The right side of the figure illustrates an alternative case in which the picture contains two slices in the first tile and one slice in the second tile.

Unlike slices, tiles are always rectangular. A tile always contains an integer number of CTUs, and may consist of CTUs contained in more than one slice. Similarly, a slice may consist of CTUs contained in more than one tile.

One or both of the following conditions shall be fulfilled for each slice and tile:

- All CTUs in a slice belong to the same tile.
- All CTUs in a tile belong to the same slice.

NOTE 1 Within the same picture, there can be both slices that contain multiple tiles and tiles that contain multiple slices.

One or both of the following conditions shall be fulfilled for each slice segment and tile:

- All CTUs in a slice segment belong to the same tile.
- All CTUs in a tile belong to the same slice segment.

When a picture is coded using three separate colour planes (`separate_colour_plane_flag` is equal to 1), a slice contains only CTBs of one colour component being identified by the corresponding value of `colour_plane_id`, and each colour component array of a picture consists of slices having the same `colour_plane_id` value. Coded slices with different values of `colour_plane_id` within a picture may be interleaved with each other under the constraint that for each value of `colour_plane_id`, the coded slice segment NAL units with that value of `colour_plane_id` shall be in the order of increasing CTB address in tile scan order for the first CTB of each coded slice segment NAL unit.

NOTE 2 When `separate_colour_plane_flag` is equal to 0, each CTB of a picture is contained in exactly one slice. When `separate_colour_plane_flag` is equal to 1, each CTB of a colour component is contained in exactly one slice (i.e. information for each CTB of a picture is present in exactly three slices and these three slices have different values of `colour_plane_id`).

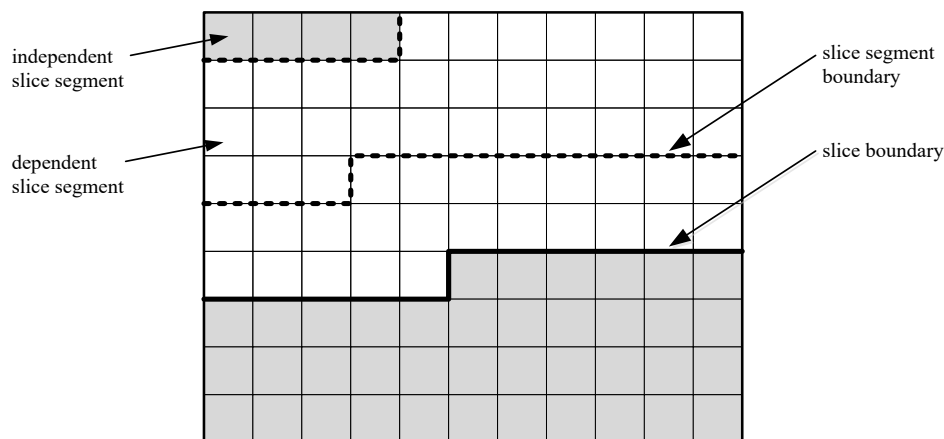


Figure 6-4 — A picture with 11 by 9 luma CTBs that is partitioned into two slices, the first of which is partitioned into three slice segments

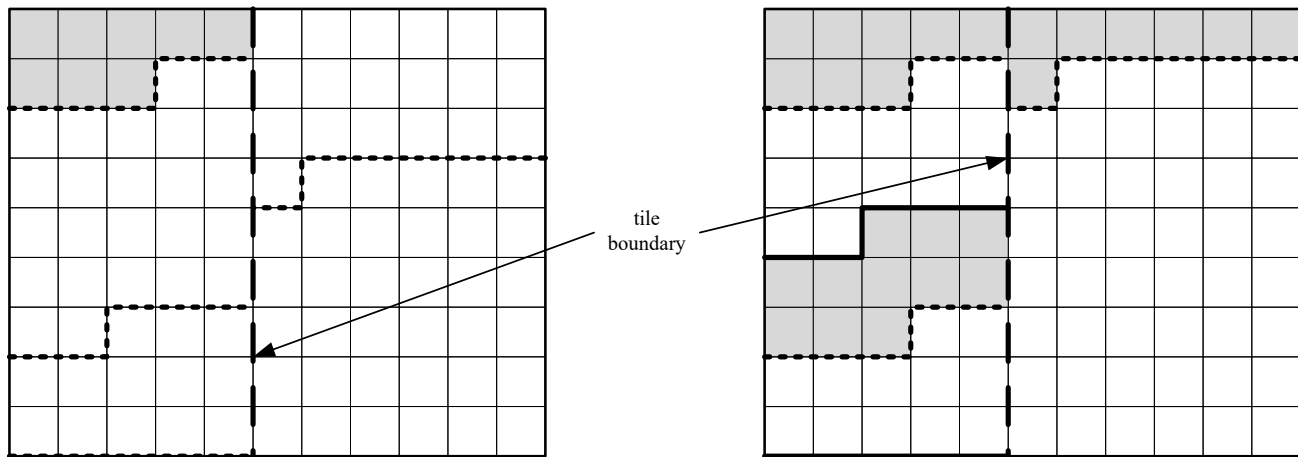


Figure 6-5 — A picture with 11 by 9 luma CTBs that is partitioned into two tiles and one slice (left) or is partitioned into two tiles and three slices (right)

6.3.2 Block and quadtree structures

The samples are processed in units of CTBs. The array size for each luma CTB in both width and height is `CtbSizeY` in units of samples. The width and height of the array for each chroma CTB are `CtbWidthC` and `CtbHeightC`, respectively, in units of samples.

Each CTB is assigned a partition signalling to identify the block sizes for intra or inter prediction and for transform coding. The partitioning is a recursive quadtree partitioning. The root of the quadtree is associated with the CTB. The quadtree is split until a leaf is reached, which is referred to as the coding block. When the component width is not an integer number of the CTB size, the CTBs at the right component boundary are incomplete. When the component height is not an integer multiple of the CTB size, the CTBs at the bottom component boundary are incomplete.

The coding block is the root node of two trees, the prediction tree and the transform tree. The prediction tree specifies the position and size of prediction blocks. The transform tree specifies the position and size of transform blocks. The splitting information for luma and chroma is identical for the prediction tree and may or may not be identical for the transform tree.

The blocks and associated syntax structures are grouped into “unit” structures as follows:

- One prediction block (monochrome picture or `separate_colour_plane_flag` is equal to 1) or three prediction blocks (luma and chroma components of a picture in 4:2:0 or 4:4:4 colour format) or five prediction blocks (luma and chroma components of a picture in 4:2:2 colour format) and the associated prediction syntax structures units are associated with a prediction unit.
- One transform block (monochrome picture or `separate_colour_plane_flag` is equal to 1) or three transform blocks (luma and chroma components of a picture in 4:2:0 or 4:4:4 colour format) or five transform blocks (luma and chroma components of a picture in 4:2:2 colour format) and the associated transform syntax structures units are associated with a transform unit.
- One coding block (monochrome picture or `separate_colour_plane_flag` is equal to 1) or three coding blocks (luma and chroma), the associated coding syntax structures and the associated prediction and transform units are associated with a coding unit.
- One CTB (monochrome picture or `separate_colour_plane_flag` is equal to 1) or three CTBs (luma and chroma), the associated coding tree syntax structures and the associated coding units are associated with a CTU.

6.3.3 Spatial or component-wise partitionings

The following divisions of processing elements of this document form spatial or component-wise partitionings:

- The division of each picture into components
- The division of each component into CTBs
- The division of each picture into tile columns
- The division of each picture into tile rows
- The division of each tile column into tiles
- The division of each tile row into tiles
- The division of each tile into CTUs
- The division of each picture into slices
- The division of each slice into slice segments
- The division of each slice segment into CTUs
- The division of each CTU into CTBs
- The division of each CTB into coding blocks, except that the CTBs are incomplete at the right component boundary when the component width is not an integer multiple of the CTB size and the CTBs are incomplete at the bottom component boundary when the component height is not an integer multiple of the CTB size
- The division of each CTU into coding units, except that the CTUs are incomplete at the right picture boundary when the picture width in luma samples is not an integer multiple of the luma CTB size and the CTUs are incomplete at the bottom picture boundary when the picture height in luma samples is not an integer multiple of the luma CTB size
- The division of each coding unit into prediction units
- The division of each coding unit into transform units
- The division of each coding unit into coding blocks
- The division of each coding block into prediction blocks
- The division of each coding block into transform blocks
- The division of each prediction unit into prediction blocks
- The division of each transform unit into transform blocks.

6.4 Availability processes

6.4.1 Derivation process for z-scan order block availability

Inputs to this process are

- the luma location (x_{Curr} , y_{Curr}) of the top-left sample of the current block relative to the top-left luma sample of the current picture, and
- the luma location (x_{NbY} , y_{NbY}) covered by a neighbouring block relative to the top-left luma sample of the current picture.

Output of this process is the availability of the neighbouring block covering the location (x_{NbY} , y_{NbY}), denoted as $availableN$.

The minimum luma block address in z-scan order $minBlockAddrCurr$ of the current block is derived as follows:

$$minBlockAddrCurr = MinTbAddrZs[x_{Curr} \gg MinTbLog2SizeY][y_{Curr} \gg MinTbLog2SizeY] \quad (6-1)$$

The minimum luma block address in z-scan order $minBlockAddrN$ of the neighbouring block covering the location (x_{NbY} , y_{NbY}) is derived as follows.

- If one or more of the following conditions are true, $minBlockAddrN$ is set equal to -1:

- x_{NbY} is less than 0;
- y_{NbY} is less than 0;
- x_{NbY} is greater than or equal to $pic_width_in_luma_samples$;
- y_{NbY} is greater than or equal to $pic_height_in_luma_samples$.

- Otherwise (x_{NbY} and y_{NbY} are inside the picture boundaries),

$$minBlockAddrN = MinTbAddrZs[x_{NbY} \gg MinTbLog2SizeY][y_{NbY} \gg MinTbLog2SizeY] \quad (6-2)$$

The neighbouring block availability $availableN$ is derived as follows.

- If one or more of the following conditions are true, $availableN$ is set equal to FALSE:

- $minBlockAddrN$ is less than 0;
- $minBlockAddrN$ is greater than $minBlockAddrCurr$;
- the variable $SliceAddrRs$ associated with the slice segment containing the neighbouring block with the minimum luma block address $minBlockAddrN$ differs in value from the variable $SliceAddrRs$ associated with the slice segment containing the current block with the minimum luma block address $minBlockAddrCurr$;
- the neighbouring block with the minimum luma block address $minBlockAddrN$ is contained in a different tile than the current block with the minimum luma block address $minBlockAddrCurr$.

- Otherwise, $availableN$ is set equal to TRUE.

6.4.2 Derivation process for prediction block availability

Inputs to this process are

- the luma location (x_{Cb} , y_{Cb}) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,