
**Information technology — MPEG
systems technologies —**

**Part 11:
Energy-efficient media consumption
(green metadata)**

*Technologies de l'information — Technologies des systèmes MPEG —
Partie 11: Consommation des supports éconergétiques (métadonnées
vertes)*





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	v
Introduction	vi
1 Scope	1
2 Normative references	1
3 Terms, definitions, symbols, abbreviated terms	1
3.1 Terms and definitions	1
3.2 Symbols and abbreviated terms	2
3.2.1 Symbols	2
3.2.2 Abbreviations	3
4 Conventions	4
4.1 Arithmetic operators	4
4.2 Relational operators	4
4.3 Bit-wise operators	4
4.4 Assignment operators	5
4.5 Range notation	5
4.6 Mathematical functions	5
4.7 Specification of syntax functions and descriptors	6
5 Functional architecture	6
5.1 Description of the functional architecture	6
5.2 Definition of components in the functional architecture	7
6 Decoder power reduction	8
6.1 General	8
6.2 Complexity metrics for decoder-power reduction	8
6.2.1 General	8
6.2.2 Syntax	8
6.2.3 Signalling	12
6.2.4 Semantics	12
6.3 Interactive signalling for remote decoder-power reduction	35
6.3.1 General	35
6.3.2 Syntax	35
6.3.3 Signalling	36
6.3.4 Semantics	36
7 Display power reduction using display adaptation	37
7.1 General	37
7.2 Syntax	37
7.2.1 Systems without a signalling mechanism from the receiver to the transmitter	37
7.2.2 Systems with a signalling mechanism from the receiver to the transmitter	38
7.3 Signalling	38
7.3.1 Systems without a signalling mechanism from the receiver to the transmitter	38
7.3.2 Systems with a signalling mechanism from the receiver to the transmitter	38
7.4 Semantics	38
8 Energy-efficient media selection	40
8.1 General	40
8.2 Syntax	40
8.3 Signalling	40
8.4 Semantics	41
8.4.1 Decoder-power indication metadata semantics	41
8.4.2 Display-power indication metadata semantics	41
9 Metrics for quality recovery after low-power encoding	42

9.1	General.....	42
9.2	Syntax.....	42
9.2.1	AVC and HEVC syntax.....	42
9.2.2	VVC syntax.....	42
9.3	Signalling.....	42
9.4	Semantics.....	43
9.4.1	AVC and HEVC Semantics.....	43
9.4.2	VVC Semantics.....	43
10	Conformance and reference software.....	44
Annex A (normative)	Supplemental Enhancement Information (SEI) syntax.....	45
Annex B (informative)	Implementation guidelines for the usage of green metadata.....	51
Annex C (normative)	Conformance and reference software.....	73
Annex D (informative)	Objective distortion metrics.....	78
Bibliography		83

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).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see <https://patents.iec.ch>).

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*.

This third edition cancels and replaces the second edition (ISO/IEC 23001-11:2019), which has been technically revised.

The main changes are as follows:

- [6.2](#) related to complexity metrics for decoder-power reduction is amended by the specification of a new VVC SEI message carrying complexity metrics for decoder-power reduction.
- [Clause 9](#) related to metrics for quality recovery after low-power encoding is amended by the specification of additional metrics for quality recovery after low-power encoding in the newly added VVC SEI message.
- [6.3](#) related to interactive signalling for remote decoder-power reduction is amended by adding new syntax elements allowing a finer control by decoder of the encoding operations.

A list of all parts in the ISO/IEC 23001 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

This document specifies the metadata (green metadata) that facilitates reduction of energy usage during media consumption as follows:

- the format of the metadata that enables reduced decoder power consumption;
- the format of the metadata that enables reduced display power consumption;
- the format of the metadata that enables media selection for joint decoder and display power reduction;
- the format of the metadata that enables quality recovery after low-power encoding.

This metadata facilitates reduced energy usage during media consumption without any degradation in the quality of experience (QoE). However, it is also possible to use this metadata to get larger energy savings, but at the expense of some QoE degradation.

The metadata for energy-efficient decoding specifies two sets of information: complexity metrics (CM) metadata and decoding operation reduction request (DOR-Req) metadata. A decoder uses CM metadata to vary operating frequency and thus reduce decoder power consumption. In a point-to-point video conferencing application, the remote encoder uses the DOR-Req metadata to modify the decoding complexity of the bitstream and thus reduce local decoder power consumption.

The metadata for energy-efficient encoding specifies quality metrics that are used by a decoder to reduce the quality loss from low-power encoding.

The metadata for energy-efficient presentation specifies RGB-component statistics and quality levels. A presentation subsystem uses this metadata to reduce power by adjusting display parameters, based on the statistics, to provide a desired quality level from those provided in the metadata.

The metadata for energy-efficient media selection specifies DOR-Req parameters, RGB-component statistics and quality levels. The client in an adaptive streaming session uses this metadata to determine decoder and display power-saving characteristics of available video representations and to select the representation with the optimal quality for a given power-saving.

Information technology — MPEG systems technologies —

Part 11:

Energy-efficient media consumption (green metadata)

1 Scope

This document specifies metadata for energy-efficient decoding, encoding, presentation, and selection of media.

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/IEC 14496-10, *Information technology — Coding of audio-visual objects — Part 10: Advanced video coding*

ISO/IEC 23008-2, *Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding*

ISO/IEC 23009-1, *Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats*

ISO/IEC 23090-3, *Information technology — Coded representation of immersive media — Part 3: Versatile video coding*

3 Terms, definitions, symbols, abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14496-10, ISO/IEC 23008-2, ISO/IEC 23009-1, ISO/IEC 23090-3 and the following 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/>

3.1.1

alpha-point deblocking instance

APDI

single filtering operation that produces either a single, filtered output p'_0 or a single, filtered output q'_0 , where p'_0 and q'_0 are filtered samples across a 4x4 block edge

3.1.2

deblocking filtering instance

single filtering operation that produces either a single, filtered output p' or a single, filtered output q' , where p' and q' are filtered samples across an 8x8 and 4x4 block edge for HEVC and VVC, respectively

3.1.3

decoding process

process that reads a bitstream and derives decoded pictures from it

Note 1 to entry: The decoding process is specified in ISO/IEC 14496-10, ISO/IEC 23008-2 or ISO/IEC 23090-3.

3.1.4

encoding process

process that produces a bitstream

Note 1 to entry: The bitstream shall conform to ISO/IEC 14496-10, ISO/IEC 23008-2 or ISO/IEC 23090-3.

3.1.5

no-quality-loss operating point

NQLOP

metadata-enabled operating point associated with the largest display-power reduction that can be achieved without any quality loss (infinite PSNR)

3.1.6

non-zero block

block containing at least one non-zero transform coefficient

3.1.7

peak signal

maximum permissible RGB component in a reconstructed frame

Note 1 to entry: For N -bit video, peak signal is $(2^N - 1)$.

3.1.8

period

interval over which complexity-metrics metadata are applicable

3.1.9

pixel

smallest addressable element in an all-points addressable display device

3.1.10

reconstructed frames

frames obtained after applying RGB colour-space conversion and cropping to the specific decoded picture or pictures for which display power-reduction metadata are applicable

3.1.11

RGB component

single sample representing one of the three primary colours of the RGB colour space

3.1.13

six-tap filtering

STF

single application of the 6-tap filter to generate a single filtered sample for fractional positions using the samples at integer-sample positions

3.2 Symbols and abbreviated terms

3.2.1 Symbols

+	addition
-	subtraction (as a two-argument operator) or negation (as a unary prefix operator)
*	multiplication

/	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 equations where no truncation or rounding is intended

3.2.2 Abbreviations

APDI	alpha-point deblocking instance
ASIC	application specific integrated circuit
AVC	advanced video coding – ISO/IEC 14496-10
BMFF	base media file format
CM	complexity metric
CMOS	complementary metal oxide semiconductor
CMP	cubemap projection format
CPU	central processing Unit
DASH	dynamic adaptive streaming over HTTP
DOR-Ratio	decoding operation reduction ratio
DOR-Req	decoding operation reduction request
DVFS	dynamic voltage frequency scaling
ERP	equi-rectangular projection format
Fps	frames per second
FS	fresh start
GP	good picture
HEVC	high efficiency video coding – ISO/IEC 23008-2
HCMP	hemisphere cubemap projection format
Mbps	mega bits per second
MPD	media presentation description
MSD	mean square difference
MV	motion vector
NQLOP	no-quality-loss operating point
PSNR	peak signal-to-noise ratio
QoE	quality of experience
RBLL	remaining battery life level
RGB	red, green, blue

SEI	supplemental enhancement information
SP	start picture
STF	six-tap filtering
SSIM	structural similarity index measure
VVC	versatile video coding – ISO/IEC 23090-3
XSD	cross-segment decoding
wPSNR	weighted peak signal-to-noise ratio
WS-PSNR	weighted to spherically uniform peak signal-to-noise ratio

4 Conventions

4.1 Arithmetic operators

x^y	exponentiation
x/y	division where no truncation or rounding is intended
	division 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
$\sum_{p \text{ in } B}^y f(p)$	summation of $f(p)$ with p taking all integer location values in a block B in a picture
$x \% y$	Modulus. Remainder of x divided by y , defined only for integers x and y with $x \geq 0$ and $y > 0$

4.2 Relational operators

$>$	greater than
$\geq$	greater than or equal to
$<$	less than
$\leq$	less than or equal to
$=$	equal to
$\neq$	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.

4.3 Bit-wise operators

$x \gg y$	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 most significant bits (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 least significant bits (LSBs) as a result of the left shift have a value equal to 0.

4.4 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 $x = x - (-3)$

4.5 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

4.6 Mathematical functions

Mathematical functions are defined as follows:

$$\text{Abs}(x) = \begin{cases} -x, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (4-1)$$

$$\text{Clip}(x) = \begin{cases} x, & x < 256 \\ 255, & \text{otherwise} \end{cases} \quad (4-2)$$

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; & z < x \\ y & ; & z > y \\ z & ; & \text{otherwise} \end{cases} \quad (4-3)$$

$$\text{Floor}(x) \text{ is the greatest integer less than or equal to } x \quad (4-4)$$

$$\text{Log}_{10}(x) \text{ returns the base-10 logarithm of } x \quad (4-5)$$

$$\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5) \quad (4-6)$$

$$\text{Sign}(x) = \begin{cases} -1, & x < 0 \\ 1, & x \geq 0 \end{cases} \quad (4-7)$$

$$x^y \text{ specifies } x \text{ to the power of } y \quad (4-8)$$

power(*x*, *y*) specifies *x* to the power of *y*

(4-9)

4.7 Specification of syntax functions and descriptors

The following function is used in the specification of the syntax:

read_bits(*n*) reads the next *n* bits from the bitstream and advances the bitstream pointer by *n* bit positions. When *n* is equal to 0, read_bits(*n*) is specified to return a value equal to 0 and to not advance the bitstream pointer.

The following descriptors specify the parsing process of each syntax element:

- u(*n*): unsigned integer using *n* bits. The parsing process for this descriptor is specified by the return value of the function read_bits(*n*) interpreted as a binary representation of an unsigned integer with most significant bit written first.
- s(*n*): signed integer using *n* bits. The parsing process for this descriptor is specified by the return value of the function read_bits(*n*) interpreted as a two's complement integer representation with most significant bit written first.

5 Functional architecture

5.1 Description of the functional architecture

Figure 1 shows the functional architecture utilizing green metadata. The media pre-processor is applied to analyse and to filter the content source and a video encoder is used to encode the content to a bitstream for delivery. The bitstream is delivered to the receiver and decoded by a video decoder with the output rendered on a presentation subsystem that implements a display process.

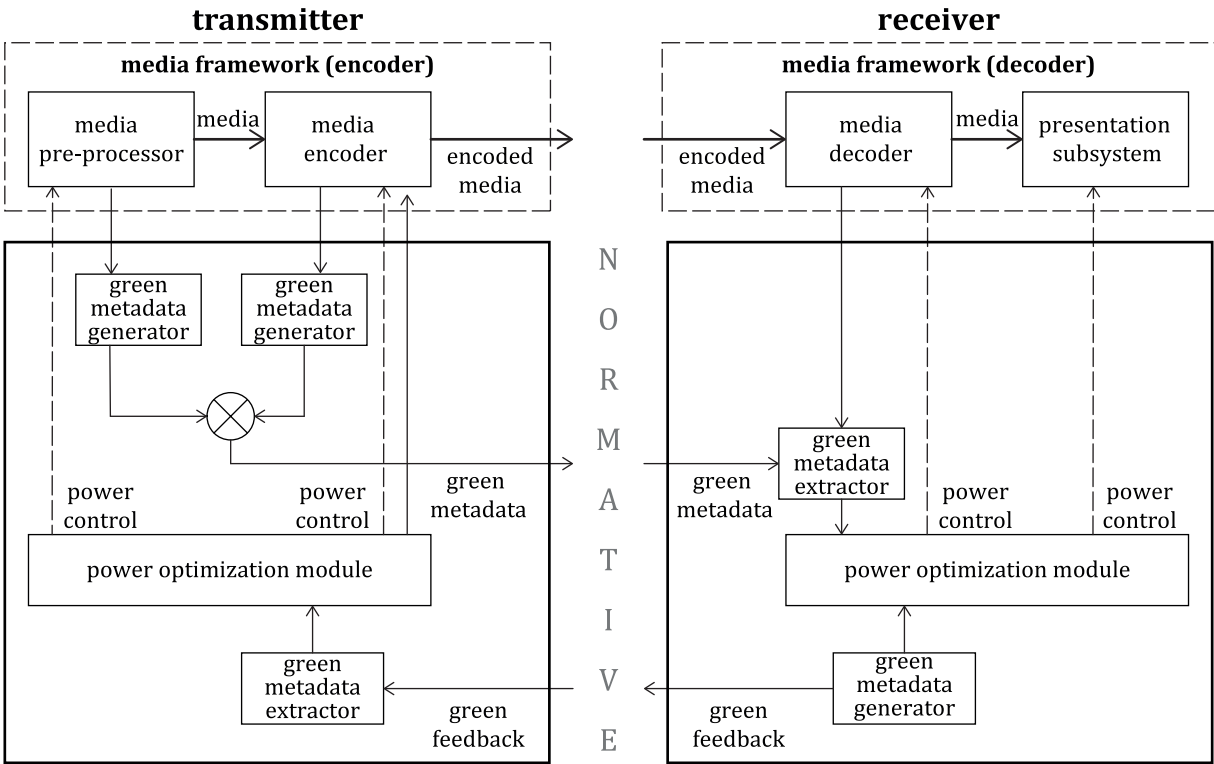


Figure 1 — Functional architecture

The green metadata is extracted from either the media encoder or the media pre-processor. In both cases, the green metadata is multiplexed or encapsulated in the conformant bitstream. Such green metadata is used at the receiver to reduce the power consumption for video decoding and presentation. The bitstream is packetized and delivered to the receiver for decoding and presentation. At the receiver, the metadata extractor processes the packets and sends the green metadata to a power optimization module for efficient power control. For instance, the power optimization module interprets the green metadata and then applies appropriate operations to reduce the video decoder's power consumption when decoding the video and to reduce the presentation subsystem's power consumption when rendering the video. In addition, the power-optimization module can collect receiver information, such as remaining battery capacity, and send it to the transmitter as green feedback to adapt the encoder operations for power-consumption reduction.

The normative aspect of this document is limited to the green metadata and green feedback in [Figure 1](#).

5.2 Definition of components in the functional architecture

green metadata generator

- Generates metadata from either the video encoder or the content pre-processor.

green metadata extractor

- Interprets the bitstream syntax information and sends it to the power optimization module in the receiver.

green feedback generator

- Generates feedback information for the transmitter.
- Communicates with the transmitter through a feedback channel, if available, for energy-efficient processing.

green feedback extractor

- Receives the feedback from the receiver and sends it to the power optimization module in the transmitter.

power optimization module in the transmitter

- Collects platform statistics such as the remaining battery capacity of the device in which the transmitter resides.
- Controls the operation of the green metadata generator, video encoder and content pre-processor.
- Processes green feedback.

power optimization module in the receiver

- Processes the green-metadata information and applies appropriate operations for power-consumption control.
- Collects platform statistics such as remaining battery capacity of the device in which the receiver resides.
- Sends requests to green feedback generator.

6 Decoder power reduction

6.1 General

Energy-efficient decoding is achieved with two types of metadata: complexity metrics (CMs) metadata and decoding operation reduction request (DOR-Req) metadata. A decoder may use CMs metadata to vary operating frequency and thus reduce decoder power consumption. In a point-to-point video conferencing application, the remote encoder may use the DOR-Req metadata to modify the decoding complexity of the bitstream and thus reduce local decoder power consumption.

6.2 Complexity metrics for decoder-power reduction

6.2.1 General

With respect to the functional architecture in [Figure 1](#), the green-metadata generator provides CMs that indicate the picture-decoding complexity of an AVC, HEVC or VVC bitstream to the decoder.

6.2.2 Syntax

The syntax for the AVC CMs is described in [Table 1](#).

Table 1 — Syntax for the AVC CMs

	Descriptor
period_type	u(8)
if ((period_type == 2) (period_type == 7)) {	
num_seconds	u(16)
}	
else if ((period_type == 3) (period_type == 8)) {	
num_pictures	u(16)
}	
if (period_type == 8) {	
temporal_map	u(8)
for (t=0; t<8; t++) {	
if ((temporal_map>>t)%2 == 1)	
num_pictures_in_temporal_layers[t]	u(16)
}	
}	
if (period_type <= 3) {	
portion_non_zero_8x8_blocks	u(8)
portion_intra_predicted_macroblocks	u(8)
portion_six_tap_filterings	u(8)
portion_alpha_point_deblocking_instances	u(8)
}	
else if (period_type == 4) {	
for (i=0; i<= num_slice_groups_minus1; i++) {	
num_slices_minus1[i]	u(16)
}	
for (i=0; i<= num_slice_groups_minus1; i++) {	
for (j=0; j<=num_slices_minus1[i]; j++) {	
first_mb_in_slice[i][j]	u(16)
portion_non_zero_8x8_blocks[i][j]	u(8)

Table 1 (continued)

<code>portion_intra_predicted_macroblocks[i][j]</code>	u(8)
<code>portion_six_tap_filterings[i][j]</code>	u(8)
<code>portion_alpha_point_deblocking_instances[i][j]</code>	u(8)
<code>}</code>	
<code>}</code>	
<code>}</code>	
<code>else if (period_type >= 5) && (period_type <= 8) {</code>	
<code>num_layers_minus1</code>	u(16)
<code>for (l=0; l<= num_layers_minus1; l++) {</code>	
<code>picture_parameter_set_id[l]</code>	u(8)
<code>priority_id[l]</code>	u(6)
<code>dependency_id[l]</code>	u(3)
<code>quality_id[l]</code>	u(4)
<code>temporal_id[l]</code>	u(3)
<code>portion_non_zero_8x8_blocks[l]</code>	u(8)
<code>portion_intra_predicted_macroblocks[l]</code>	u(8)
<code>portion_six_tap_filterings[l]</code>	u(8)
<code>portion_alpha_point_deblocking_instances[l]</code>	u(8)
<code>}</code>	
<code>}</code>	

The syntax for the HEVC CMs is described in [Table 2](#).

Table 2 — Syntax for the HEVC CMs

	Descriptor
<code>period_type</code>	u(8)
<code>if (period_type == 2) {</code>	
<code>num_seconds</code>	u(16)
<code>}</code>	
<code>else if (period_type == 3) {</code>	
<code>num_pictures</code>	u(16)
<code>}</code>	
<code>if (period_type <= 3) {</code>	
<code>portion_non_zero_blocks_area</code>	u(8)
<code>if (portion_non_zero_blocks_area != 0) {</code>	
<code>portion_8x8_blocks_in_non_zero_area</code>	u(8)
<code>portion_16x16_blocks_in_non_zero_area</code>	u(8)
<code>portion_32x32_blocks_in_non_zero_area</code>	u(8)
<code>}</code>	
<code>portion_intra_predicted_blocks_area</code>	u(8)
<code>if (portion_intra_predicted_blocks_area == 255) {</code>	
<code>portion_planar_blocks_in_intra_area</code>	u(8)
<code>portion_dc_blocks_in_intra_area</code>	u(8)
<code>portion_angular_hv_blocks_in_intra_area</code>	u(8)
<code>}</code>	
<code>else {</code>	
<code>portion_blocks_a_c_d_n_filterings</code>	u(8)

Table 2 (continued)

<code>portion_blocks_h_b_filterings</code>	<code>u(8)</code>
<code>portion_blocks_f_i_k_q_filterings</code>	<code>u(8)</code>
<code>portion_blocks_j_filterings</code>	<code>u(8)</code>
<code>portion_blocks_e_g_p_r_filterings</code>	<code>u(8)</code>
<code>}</code>	
<code>portion_deblocking_instances</code>	<code>u(8)</code>
<code>}</code>	
<code>else if (period_type == 4) {</code>	
<code> max_num_slices_tiles_minus1</code>	<code>u(16)</code>
<code> for (t=0; t<=max_num_slices_tiles_minus1; t++) {</code>	
<code> first_ctb_in_slice_or_tile[t]</code>	<code>u(16)</code>
<code> portion_non_zero_blocks_area[t]</code>	<code>u(8)</code>
<code> if (portion_non_zero_blocks_area[t] != 0) {</code>	
<code> portion_8x8_blocks_in_non_zero_area[t]</code>	<code>u(8)</code>
<code> portion_16x16_blocks_in_non_zero_area[t]</code>	<code>u(8)</code>
<code> portion_32x32_blocks_in_non_zero_area[t]</code>	<code>u(8)</code>
<code> }</code>	
<code> portion_intra_predicted_blocks_area[t]</code>	<code>u(8)</code>
<code> if (portion_intra_predicted_blocks_area[t] == 255) {</code>	
<code> portion_planar_blocks_in_intra_area[t]</code>	<code>u(8)</code>
<code> portion_dc_blocks_in_intra_area[t]</code>	<code>u(8)</code>
<code> portion_angular_hv_blocks_in_intra_area[t]</code>	<code>u(8)</code>
<code> }</code>	
<code> else {</code>	
<code> portion_blocks_a_c_d_n_filterings[t]</code>	<code>u(8)</code>
<code> portion_blocks_h_b_filterings[t]</code>	<code>u(8)</code>
<code> portion_blocks_f_i_k_q_filterings[t]</code>	<code>u(8)</code>
<code> portion_blocks_j_filterings[t]</code>	<code>u(8)</code>
<code> portion_blocks_e_g_p_r_filterings[t]</code>	<code>u(8)</code>
<code> }</code>	
<code> portion_deblocking_instances[t]</code>	<code>u(8)</code>
<code>}</code>	
<code>}</code>	

The syntax for the VVC CMs is described in [Table 3](#).

Table 3 — Syntax for the VVC CMs

	Descriptor
<code>period_type</code>	<code>u(4)</code>
<code>granularity_type</code>	<code>u(3)</code>
<code>extended_representation_flag</code>	<code>u(1)</code>
<code>if (period_type == 2) {</code>	
<code> num_seconds</code>	<code>u(16)</code>
<code>}</code>	
<code>else if (period_type == 3) {</code>	
<code> num_pictures</code>	<code>u(16)</code>
<code>}</code>	

Table 3 (continued)

if (granularity_type == 0) {	
portion_non_zero_blocks_area	u(8)
portion_non_zero_transform_coefficients_area	u(8)
portion_intra_predicted_blocks_area	u(8)
portion_deblocking_instances	u(8)
portion_alf_filtered_blocks	u(8)
if (extended_representation_flag) {	
if (portion_non_zero_blocks_area != 0) {	
portion_non_zero_4_8_16_blocks_area	u(8)
portion_non_zero_32_64_128_blocks_area	u(8)
portion_non_zero_256_512_1024_blocks_area	u(8)
portion_non_zero_2048_4096_blocks_area	u(8)
}	
if (portion_intra_predicted_blocks_area < 255) {	
portion_bi_and_gpm_predicted_blocks_area	u(8)
portion_bdof_blocks_area	u(8)
}	
portion_sao_filtered_blocks	u(8)
}	
}	
else if (granularity_type <= 3) {	
max_num_segments_minus1	u(16)
for (t=0; t<=max_num_segments_minus1; t++) {	
segment_address[t]	u(16)
portion_non_zero_blocks_area[t]	u(8)
portion_non_zero_transform_coefficients_area[t]	u(8)
portion_intra_predicted_blocks_area[t]	u(8)
portion_deblocking_instances[t]	u(8)
portion_alf_filtered_blocks[t]	u(8)
if (extended_representation_flag) {	
if (portion_non_zero_blocks_area[t] != 0) {	
portion_non_zero_4_8_16_blocks_area[t]	u(8)
portion_non_zero_32_64_128_blocks_area[t]	u(8)
portion_non_zero_256_512_1024_blocks_area[t]	u(8)
portion_non_zero_2048_4096_blocks_area[t]	u(8)
}	
if (portion_intra_predicted_blocks_area[t] < 255) {	
portion_bi_and_gpm_predicted_blocks_area[t]	u(8)
portion_bdof_blocks_area[t]	u(8)
}	
portion_sao_filtered_blocks[t]	u(8)
}	
}	
}	

6.2.3 Signalling

SEI messages can be used to signal green metadata in an AVC, HEVC or VVC stream. The green metadata SEI message payload type is specified in ISO/IEC 14496-10, ISO/IEC 23008-2, and ISO/IEC 23090-3. The complete syntax of the green metadata SEI message payload is specified in [Annex A](#).

The message containing the CMs is transmitted at the start of an upcoming period. The next message containing CMs is transmitted at the start of the next upcoming period. Therefore, when the upcoming period is a picture or the interval up to the next I-slice, a message is transmitted for each picture or interval, respectively. However, when the upcoming period is a specified time interval or a specified number of pictures, the associated message is transmitted with the first picture in the time interval or with the first picture in the specified number of pictures.

6.2.4 Semantics

6.2.4.1 AVC semantics

The semantics of various terms are defined below.

`period_type` specifies the type of upcoming period over which the complexity metrics are applicable and is defined in the [Table 4](#).

Table 4 — specification of `period_type` for AVC

Value	Description
0x00	complexity metrics are applicable to a single picture
0x01	complexity metrics are applicable to all pictures in decoding order, up to (but not including) the picture containing the next I slice
0x02	complexity metrics are applicable over a specified time interval in seconds
0x03	complexity metrics are applicable over a specified number of pictures counted in decoding order
0x04	complexity metrics are applicable to a single picture with slice granularity
0x05	complexity metrics are applicable to a single picture with scalable layer granularity
0x06	complexity metrics are applicable to all pictures in decoding order, up to (but not including) the picture containing the next I slice in the base layer with scalable layer granularity
0x07	complexity metrics are applicable over a specified time interval in seconds with scalable layer granularity
0x08	complexity metrics are applicable over a specified number of pictures counted in decoding order with scalable layer granularity
0x09–0xFF	user-defined

`num_seconds` indicates the number of seconds over which the complexity metrics are applicable when `period_type` is 2 or 7.

`num_pictures` indicates the number of pictures, counted in decoding order, over which the complexity metrics are applicable when `period_type` is 3 or 8. When `period_type` is 8, this is a default number of pictures for each temporal layer, which can be overridden using `temporal_map` flags.

$N_{\text{picsInPeriod}}$ specifies the number of pictures in the specified period. When `period_type` is 0 or 4, then $N_{\text{picsInPeriod}}$ is 1. When `period_type` is 1, then $N_{\text{picsInPeriod}}$ is determined by counting the pictures in decoding order up to (but not including) the one containing the next I slice. When `period_type` is 2, then $N_{\text{picsInPeriod}}$ is determined from the frame rate. When `period_type` is 3, then $N_{\text{picsInPeriod}}$ is equal to `num_pictures`.

$N_{\text{mbsInPeriod}}$ specifies the total number of macroblocks that are coded in the specified period. It is determined by the following computation:

$$N_{\text{mbsInPeriod}} = \sum_{n=1}^{N_{\text{picsInPeriod}}} N_{\text{mbsInPic}}(n) \quad (6-1)$$

where $N_{\text{mbsInPic}}(n)$ is set to the value of the AVC variable PicSizeInMbs for the n^{th} picture within the specified period, where $1 \leq n \leq N_{\text{picsInPeriod}}$.

temporal_map indicates which temporal layer has a different number of pictures from num_pictures in the specified period, when period_type is 8.

$\text{num_pictures_in_temporal_layers}[t]$ indicates the number of pictures in the specified period for the t^{th} temporal layer when period_type is 8. When not present, it is equal to num_pictures .

$N_{\text{picsInPeriodForTempLayer}}[t]$ specifies the number of pictures in the specified period for the t^{th} temporal layer. When period_type is 5 then $N_{\text{picsInPeriodForTempLayer}}[t]$ is 1. When period_type is 6, then $N_{\text{picsInPeriodForTempLayer}}[t]$ is determined by counting the pictures associated to the t^{th} temporal layer in decoding order up to (but not including) the one containing the next I slice. When period_type is 7, then $N_{\text{picsInPeriodForTempLayer}}[t]$ is determined from the frame rate associated to the t^{th} temporal layer. When period_type is 8, then $N_{\text{picsInPeriodForTempLayer}}[t]$ is equal to $\text{num_pictures_in_temporal_layers}[t]$.

$\text{portion_non_zero_8x8_blocks}$ indicates the portion of 8x8 blocks with non-zero transform coefficients values in the specified period and is set equal to $P_{\text{nonZero8x8Blks}}$ defined as follows:

$$P_{\text{nonZero8x8Blks}} = \text{Floor} \left(\frac{N_{\text{nonZero8x8Blks}}}{4 * N_{\text{mbsInPeriod}}} * 255 \right) \quad (6-2)$$

where $N_{\text{nonZero8x8Blks}}$ is the number of 8x8 blocks with non-zero transform coefficients values in the specified period. $N_{\text{nonZero8x8Blks}}$ is derived from $\text{portion_non_zero_8x8_blocks}$ and $N_{\text{mbsInPeriod}}$ in the decoder.

$\text{portion_intra_predicted_macroblocks}$ indicates the portion of intra-predicted macroblocks in the specified period and is set equal to P_{intraMbs} defined as follows:

$$P_{\text{intraMbs}} = \text{Floor} \left(\frac{N_{\text{intraMbs}}}{N_{\text{mbsInPeriod}}} * 255 \right) \quad (6-3)$$

where N_{intraMbs} is the number of intra-predicted macroblocks in the specified period. N_{intraMbs} is derived from $\text{portion_intra_predicted_macroblocks}$ and $N_{\text{mbsInPeriod}}$ in the decoder.

$\text{portion_six_tap_filterings}$ indicates the portion of 6-tap filterings (STFs) in the specified period and is set equal to $P_{\text{sixTapFilt}}$ defined as follows:

$$P_{\text{sixTapFilt}} = \text{Floor} \left(\frac{N_{\text{sixTapFilt}}}{N_{\text{maxSixTapFiltInPeriod}}} * 255 \right) \quad (6-4)$$

where $N_{\text{maxSixTapFiltInPeriod}}$ is the maximum number of STFs that can occur within the specified period and is derived from $N_{\text{mbsInPeriod}}$ variable as

$$N_{\text{maxSixTapFiltInPeriod}} = (1664 * N_{\text{mbsInPeriod}}) \quad (6-5)$$

and $N_{\text{sixTapFilt}}$ is the number of 6-tap filterings (STFs) within the specified period. Guidance for the counting of $N_{\text{sixTapFilt}}$ can be found in [Annex B](#). $N_{\text{sixTapFilt}}$ is derived from $\text{portion_six_tap_filterings}$ and $N_{\text{maxSixTapFiltInPeriod}}$ in the decoder.

portion_alpha_point_deblocking_instances indicates the portion of alpha-point deblocking instances (APDIs) in the specified period and is set equal to $P_{\text{alphaPtDbfs}}$ defined as follows:

$$P_{\text{alphaPtDbfs}} = \text{Floor} \left(\frac{N_{\text{alphaPtDbfs}}}{N_{\text{maxAlphaPtDbfsInPeriod}}} * 255 \right) \quad (6-6)$$

$N_{\text{maxAlphaPtDbfsInPeriod}}$ is the maximum number of APDIs that can occur within the specified period and is derived from $N_{\text{mbsInPeriod}}$ and S_{chrMult} variables as

$$N_{\text{maxAlphaPtDbfsInPeriod}} = 128 * S_{\text{chrMult}} * N_{\text{mbsInPeriod}} \quad (6-7)$$

S_{chrMult} depends on the AVC variables separate_colour_plane_flag and chroma_format_idc as shown in the [Table 5](#).

Table 5 — specification of S_{chrMult} for AVC

S_{chrMult}	separate_colour_plane_flag	chroma_format_idc	Comment
1	0	0	monochrome
1.5	0	1	4:2:0 sampling
2	0	2	4:2:2 sampling
3	0	3	4:4:4 sampling
3	1	any value	separate colour plane

$N_{\text{alphaPtDbfs}}$ is the number of APDIs in the specified period. Using the notation in ISO/IEC 14496-10, this is equivalent to the total number of filtering operations applied to produce filtered samples of the type p'_0 or q'_0 , in the specified period. $N_{\text{alphaPtDbfs}}$ is derived from portion_alpha_point_deblocking_instances and $N_{\text{maxAlphaPtDbfsInPeriod}}$ in the decoder.

num_slices_minus1 plus 1 indicates the number of slices per slice_group in the picture.

first_mb_in_slice[i][j] indicates the first macroblock number in the slice[i][j].

$N_{\text{mbsInSlice}}[i][j]$ is the total number of macroblocks that are coded in the slice[i][j] and is determined by the following computation:

- if num_slice_groups_minus1 is equal to 0, the following process described in pseudo-code applies.

```

if (j < num_slices_minus1[0])
     $N_{\text{mbsInSlice}}[0][j] = \text{first\_mb\_in\_slice}[0][j+1] - \text{first\_mb\_in\_slice}[0][j]$ 
else
     $N_{\text{mbsInSlice}}[0][j] = \text{PicSizeInMbs} - \text{first\_mb\_in\_slice}[0][j]$ 

```

(6-8)

- otherwise (num_slice_groups_minus1 is not equal to 0), and after derivation of the macroblock to slice group map (MbToSliceGroupMap) as specified in ISO/IEC 14496-10:2022, 8.2.2.8, the following process described in pseudo-code applies:

```

k=0;
if (j < num_slices_minus1[i])
    for ( n=first_mb_in_slice[i][j]; n < first_mb_in_slice[i][j+1]; n++ )
        if ( MbToSliceGroupMap[first_mb_in_slice[i][j]] == MbToSliceGroupMap[n] )
            k++;
     $N_{\text{mbsInSlice}}[i][j] = k;$ 
else
    for ( n=first_mb_in_slice[i][j]; n < PicSizeInMbs; n++ )
        if ( MbToSliceGroupMap[first_mb_in_slice[i][j]] == MbToSliceGroupMap[n] )
            k++;

```

$$N_{\text{mbsInSlice}}[i][j] = k; \quad (6-9)$$

portion_non_zero_8x8_blocks[i][j] indicates the portion of 8x8 blocks with non-zero transform coefficients values in the slice[i][j] and is set equal to $P_{\text{nonZero8x8Blks}}[i][j]$ defined as follows:

$$P_{\text{nonZero8x8Blks}}[i][j] = \text{Floor} \left(\frac{N_{\text{nonZero8x8Blks}}[i][j]}{4 * N_{\text{mbsInSlice}}[i][j]} * 255 \right) \quad (6-10)$$

where $N_{\text{nonZero8x8Blks}}[i][j]$ is the number of 8x8 blocks with non-zero transform coefficients values in the slice[i][j]. $N_{\text{nonZero8x8Blks}}[i][j]$ is derived from portion_non_zero_8x8_blocks[i][j] and $N_{\text{mbsInSlice}}[i][j]$ in the decoder.

portion_intra_predicted_macroblocks[i][j] indicates the portion of macroblocks using intra prediction modes in the slice[i][j] and is set equal to $P_{\text{intraMbs}}[i][j]$ defined as follows:

$$P_{\text{intraMbs}}[i][j] = \text{Floor} \left(\frac{N_{\text{intraMbs}}[i][j]}{N_{\text{mbsInSlice}}[i][j]} * 255 \right) \quad (6-11)$$

where $N_{\text{intraMbs}}[i][j]$ is the number of macroblocks using intra prediction modes in the slice[i][j]. $N_{\text{intraMbs}}[i][j]$ is derived from portion_intra_predicted_macroblocks[i][j] and $N_{\text{mbsInSlice}}[i][j]$ in the decoder.

portion_six_tap_filterings[i][j] indicates the portion of 6-tap filterings (STFs) in the specified slice[i][j] and is set equal to $P_{\text{sixTapFilt}}[i][j]$ defined as follows:

$$P_{\text{sixTapFilt}}[i][j] = \text{Floor} \left(\frac{N_{\text{sixTapFilt}}[i][j]}{N_{\text{maxSixTapFiltInSlice}}[i][j]} * 255 \right) \quad (6-12)$$

where $N_{\text{maxSixTapFiltInSlice}}[i][j]$ is the maximum number of STFs that can occur in the slice[i][j] and is derived from $N_{\text{mbsInSlice}}[i][j]$ variable as

$$N_{\text{maxSixTapFiltInSlice}}[i][j] = 1664 * N_{\text{mbsInSlice}}[i][j] \quad (6-13)$$

and $N_{\text{sixTapFilt}}[i][j]$ is the number of 6-tap filterings (STFs) within the slice[i][j]. Guidance for the counting of $N_{\text{sixTapFilt}}[i][j]$ can be found in [Annex B](#). $N_{\text{sixTapFilt}}[i][j]$ is derived from portion_six_tap_filterings[i][j] and $N_{\text{maxSixTapFiltInSlice}}[i][j]$ in the decoder.

portion_alpha_point_deblocking_instances[i][j] indicates the portion of alpha-point deblocking instances (APDIs) in the specified slice[i][j] and is set equal to $P_{\text{alphaPtDbfs}}[i][j]$ defined as follows:

$$P_{\text{alphaPtDbfs}}[i][j] = \text{Floor} \left(\frac{N_{\text{alphaPtDbfs}}[i][j]}{N_{\text{maxAlphaPtDbfsInSlice}}[i][j]} * 255 \right) \quad (6-14)$$

where $N_{\text{maxAlphaPtDbfsInSlice}}[i][j]$ is the maximum number of APDIs that can occur in the slice[i][j] and is derived from $N_{\text{mbsInSlice}}[i][j]$ and S_{chrMult} variables as

$$N_{\text{maxAlphaPtDbfsInSlice}}[i][j] = 128 * S_{\text{chrMult}} * N_{\text{mbsInSlice}}[i][j] \quad (6-15)$$

and $N_{\text{alphaPtDbfs}}[i][j]$ is the number of alpha-point deblocking instances (APDIs) in slice[i][j]. $N_{\text{alphaPtDbfs}}[i][j]$ is derived from portion_alpha_point_deblocking_instances[i][j] and $N_{\text{maxAlphaPtDbfsInSlice}}[i][j]$ in the decoder.

num_layers_minus1 plus 1 indicates the number of scalable layers in the associated picture or in the specified period.

pic_parameter_set_id[l] indicates the picture parameter set in use for the l^{th} scalable layer. The value of pic_parameter_set_id[l] shall be in the range of 0 to 255, inclusive (as specified in ISO/IEC 14496-10:2022, 7.4.3).

$\text{priority_id}[l]$ indicates a priority identifier for the NAL unit in the l^{th} scalable layer. The value of $\text{priority_id}[l]$ shall be in the range of 0 to 63, inclusive (as specified in ISO/IEC 14496-10:2022, G.7.3.1.1).

$\text{dependency_id}[l]$ indicates a dependency identifier for the NAL unit in the l^{th} scalable layer. The value of $\text{dependency_id}[l]$ shall be in the range of 0 to 7, inclusive (as specified in ISO/IEC 14496-10:2022, G.7.3.1.1).

$\text{quality_id}[l]$ indicates a quality identifier for the NAL unit in the l^{th} scalable layer. The value of $\text{quality_id}[l]$ shall be in the range of 0 to 15, inclusive (as specified in ISO/IEC 14496-10:2022, G.7.3.1.1).

$\text{temporal_id}[l]$ indicates a temporal identifier for the NAL unit in the l^{th} scalable layer. The value of $\text{temporal_id}[l]$ shall be in the range of 0 to 7, inclusive (as specified in ISO/IEC 14496-10:2022, G.7.3.1.1).

$\text{portion_non_zero_8x8_blocks}[l]$ indicates the portion of 8x8 blocks with non-zero transform coefficients values in the l^{th} scalable layer and is set equal to $P_{\text{nonZero8x8Blks}}[i][j]$ defined as follows:

$$P_{\text{nonZero8x8Blks}}[l] = \text{Floor} \left(\frac{N_{\text{nonZero8x8Blks}}[l]}{4 * N_{\text{mbsInLayerInPeriod}}[l]} * 255 \right) \quad (6-16)$$

$N_{\text{mbsInLayerInPeriod}}[l]$ is the total number of macroblocks in the l^{th} scalable layer in the specified period and is derived from $N_{\text{mbsInLayerInPeriod}}[l]$ and $N_{\text{picsInPeriodForTempLayer}}[\text{temporal_id}[l]]$ as

$$N_{\text{mbsInLayerInPeriod}}[l] = N_{\text{mbsInLayer}}[l] * N_{\text{picsInPeriodForTempLayer}}[\text{temporal_id}[l]] \quad (6-17)$$

where $N_{\text{mbsInLayer}}[l]$ is the total number of macroblocks in the l^{th} scalable layer and determined after derivation of the number of macroblocks associated with $\text{pic_parameter_set_id}[l]$, as specified in ISO/IEC 14496-10:2022, 7.4.3.

$N_{\text{nonZero8x8Blks}}[l]$ is the number of 8x8 blocks with non-zero transform coefficients values in the l^{th} scalable layer in the specified period. It is derived from $\text{portion_non_zero_8x8_blocks}[l]$ and $N_{\text{mbsInLayerInPeriod}}[l]$ in the decoder.

$\text{portion_intra_predicted_macroblocks}[l]$ indicates the portion of macroblocks using intra prediction modes in the l^{th} scalable layer and is set equal to $P_{\text{intraMbs}}[i][j]$ defined as follows:

$$P_{\text{intraMbs}}[l] = \text{Floor} \left(\frac{N_{\text{intraMbs}}[l]}{N_{\text{mbsInLayerInPeriod}}[l]} * 255 \right) \quad (6-18)$$

$N_{\text{intraMbs}}[l]$ is the number of macroblocks using intra prediction modes in the l^{th} scalable layer in the specified period. It is derived from $\text{portion_intra_predicted_macroblocks}[l]$ and $N_{\text{mbsInLayerInPeriod}}[l]$ in the decoder.

$\text{portion_six_tap_filterings}[l]$ indicates the portion of 6-tap filterings (STFs) in the specified l^{th} scalable layer in the specified period and is set equal to $P_{\text{sixTapFilt}}[i][j]$ defined as follows:

$$P_{\text{sixTapFilt}}[l] = \text{Floor} \left(\frac{N_{\text{sixTapFilter}}[l]}{N_{\text{maxSixTapFiltInLayerInPeriod}}[l]} * 255 \right) \quad (6-19)$$

$N_{\text{maxSixTapFiltInLayerInPeriod}}[l]$ is the maximum number of STFs that can occur in the l^{th} scalable layer in the specified period and is derived from $N_{\text{mbsInLayerInPeriod}}[l]$ variable as

$$N_{\text{maxSixTapFiltInLayerInPeriod}}[l] = 1664 * N_{\text{mbsInLayerInPeriod}}[l] \quad (6-20)$$

$N_{\text{sixTapFilt}}[l]$ is the number of 6-tap filterings (STFs) within the l^{th} scalable layer in the specified period. Guidance for the counting of $N_{\text{sixTapFilt}}[l]$ can be found in [Annex B](#). It is derived from $\text{portion_six_tap_filterings}[l]$ and $N_{\text{maxSixTapFiltInLayerInPeriod}}[l]$ in the decoder.

portion_alpha_point_deblocking_instances[l] indicates the portion of alpha-point deblocking instances (APDIs) in the specified lth scalable layer in the specified period and is set equal to $P_{\text{alphaPtDbfs}}[i][j]$ defined as follows:

$$P_{\text{alphaPtDbfs}}[l] = \text{Floor} \left(\frac{N_{\text{alphaPtDbfs}}[l]}{N_{\text{maxAlphaPtDbfsInLayerInPeriod}}[l]} * 255 \right) \quad (6-21)$$

$N_{\text{maxAlphaPtDbfsInLayerInPeriod}}[l]$ is the maximum number of APDIs that can occur in the lth scalable layer in the specified period and is derived from $N_{\text{mbsInLayerInPeriod}}[l]$ and S_{chrMult} variables as

$$N_{\text{maxAlphaPtDbfsInLayerInPeriod}}[l] = 128 * S_{\text{chrMult}} * N_{\text{mbsInLayerInPeriod}}[l] \quad (6-22)$$

$N_{\text{alphaPtDbfs}}[l]$ is the number of alpha-point deblocking instances (APDIs) in the lth scalable layer in the specified period. It is derived from portion_alpha_point_deblocking_instances[l] and $N_{\text{maxAlphaPtDbfsInLayerInPeriod}}[l]$ in the decoder.

6.2.4.2 HEVC semantics

The semantics of various terms are defined below.

period_type specifies the type of upcoming period over which the complexity metrics are applicable and is defined in the [Table 6](#).

Table 6 — specification of period_type for HEVC

Value	Description
0x00	complexity metrics are applicable to a single picture
0x01	complexity metrics are applicable to all pictures in decoding order, up to (but not including) the picture containing the next I slice
0x02	complexity metrics are applicable over a specified time interval in seconds
0x03	complexity metrics are applicable over a specified number of pictures counted in decoding order
0x04	complexity metrics are applicable to a single picture with slice or tile granularity
0x05-0xFF	reserved

num_seconds indicates the number of seconds over which the complexity metrics are applicable when period_type is 2.

num_pictures indicates the number of pictures, counted in decoding order, over which the complexity metrics are applicable when period_type is 3.

$N_{\text{picsInPeriod}}$ is the number of pictures in the specified period. When period_type is 0, then $N_{\text{picsInPeriod}}$ is 1. When period_type is 1, then $N_{\text{picsInPeriod}}$ is determined by counting the pictures in decoding order up to (but not including) the one containing the next I slice. When period_type is 2, then $N_{\text{picsInPeriod}}$ is determined from the frame rate. When period_type is 3, then $N_{\text{picsInPeriod}}$ is equal to num_pictures.

$N_{4x4BlksInPeriod}$ is the total number of 4x4 blocks that are coded in the specified period.

It is determined by the following computation:

$$N_{4x4BlksInPeriod} = \sum_{n=1}^{N_{\text{picsInPeriod}}} N_{4x4BlksPic}(n) \quad (6-23)$$

where $N_{4x4BlksPic}(n)$ is derived for the nth picture within the specified period, with $1 \leq n \leq N_{\text{picsInPeriod}}$, from HEVC variables CtbLog2SizeY and PicSizeInCtbsY as follows:

$$N_{4x4BlksPic}(n) = S_{\text{picInCtb}} * N_{\text{ctbs}} \quad (6-24)$$

where

- N_{ctbs} is set equal to $(1 \ll (\text{CtbLog2SizeY} - 2))^2$
- S_{picInCtb} is set equal to PicSizeInCtbsY .

portion_non_zero_blocks_area indicates the portion of area covered by blocks with non-zero transform coefficients values, in the pictures of the specified period, using a 4x4 blocks granularity and is set equal to $P_{\text{nonZeroBlksArea}}$ defined as follows:

$$P_{\text{nonZeroBlksArea}} = \text{Floor} \left(\frac{N_{\text{nonZeroBlks}}}{N_{4 \times 4 \text{BlksInPeriod}}} * 255 \right) \quad (6-25)$$

where $N_{\text{nonZeroBlks}}$ is the number of blocks with non-zero transform coefficients values in the specified period using 4x4 granularity. At the encoder side, $N_{\text{nonZeroBlks}}$ is computed as follows:

$$N_{\text{nonZeroBlks}} = N_{\text{nonZero}4 \times 4 \text{Blks}} + 4 * N_{\text{nonZero}8 \times 8 \text{Blks}} + 16 * N_{\text{nonZero}16 \times 16 \text{Blks}} + 64 * N_{\text{nonZero}32 \times 32 \text{Blks}} \quad (6-26)$$

where $N_{\text{nonZero}4 \times 4 \text{Blks}}$, $N_{\text{nonZero}8 \times 8 \text{Blks}}$, $N_{\text{nonZero}16 \times 16 \text{Blks}}$, $N_{\text{nonZero}32 \times 32 \text{Blks}}$ are the number of 4x4, 8x8, 16x16 and 32x32 blocks with non-zero transform coefficients values, respectively, in the specified period.

$N_{\text{nonZeroBlks}}$ is derived from portion_non_zero_blocks_area and $N_{4 \times 4 \text{BlksInPeriod}}$ in the decoder.

portion_8x8_blocks_in_non_zero_area indicates the portion of 8x8 blocks area in the non-zero area in the specified period and is set equal to $P_{\text{nonZero}8 \times 8 \text{Blks}}$ defined as follows:

$$P_{\text{nonZero}8 \times 8 \text{Blks}} = \text{Floor} \left(\frac{4 * N_{\text{nonZero}8 \times 8 \text{Blks}}}{N_{\text{nonZeroBlks}}} * 255 \right) \quad (6-27)$$

When not present, it is set equal to 0.

$N_{\text{nonZero}8 \times 8 \text{Blks}}$ is the number of 8x8 blocks with non-zero transform coefficients values in the specified period. It is derived from portion_8x8_blocks_in_non_zero_area and $N_{\text{nonZeroBlks}}$ in the decoder.

portion_16x16_blocks_in_non_zero_area indicates the portion of 16x16 blocks area in the non-zero area in the specified period and is set equal to $P_{\text{nonZero}16 \times 16 \text{Blks}}$ defined as follows:

$$P_{\text{nonZero}16 \times 16 \text{Blks}} = \text{Floor} \left(\frac{16 * N_{\text{nonZero}16 \times 16 \text{Blks}}}{N_{\text{nonZeroBlks}}} * 255 \right) \quad (6-28)$$

When not present, is equal to 0.

$N_{\text{nonZero}16 \times 16 \text{Blks}}$ is the number of 16x16 blocks with non-zero transform coefficients values in the specified period. It is derived from portion_16x16_blocks_in_non_zero_area and $N_{\text{nonZeroBlks}}$ in the decoder.

portion_32x32_blocks_in_non_zero_area indicates the portion of 32x32 blocks area in the non-zero area in the specified period and is set equal to $P_{\text{nonZero}32 \times 32 \text{Blks}}$ defined as follows:

$$P_{\text{nonZero}32 \times 32 \text{Blks}} = \text{Floor} \left(\frac{64 * N_{\text{nonZero}32 \times 32 \text{Blks}}}{N_{\text{nonZeroBlks}}} * 255 \right) \quad (6-29)$$

When not present, it is set equal to 0.

$N_{\text{nonZero}32 \times 32 \text{Blks}}$ is the number of 32x32 blocks with non-zero transform coefficients values in the specified period. It is derived from portion_32x32_blocks_in_non_zero_area and $N_{\text{nonZeroBlks}}$ in the decoder.

$N_{\text{nonZero4x4Blks}}$ is the number of 4x4 blocks with non-zero transform coefficients values in the specified period. $N_{\text{nonZero4x4Blks}}$ is derived from $N_{\text{nonZeroBlks}}$, $N_{\text{nonZero8x8Blks}}$, $N_{\text{nonZero16x16Blks}}$ and $N_{\text{nonZero32x32Blks}}$ as follows in the decoder:

$$N_{\text{nonZero4x4Blks}} = N_{\text{nonZeroBlks}} - 4 * N_{\text{nonZero8x8Blks}} - 16 * N_{\text{nonZero16x16Blks}} - 64 * N_{\text{nonZero32x32Blks}} \quad (6-30)$$

portion_intra_predicted_blocks_area indicates the portion of area covered by intra predicted blocks in the pictures of the specified period using 4x4 granularity and is set equal to $P_{\text{intraBlks}}$ defined as follows:

$$P_{\text{intraBlks}} = \text{Floor} \left(\frac{4 * N_{\text{intraBlks}}}{N_{4x4BlksInPeriod}} * 255 \right) \quad (6-31)$$

$N_{\text{intraBlks}}$ is the number of intra predicted blocks in the specified period using 8x8 granularity. At the encoder side, it is computed as follows:

$$N_{\text{intraBlks}} = N_{\text{intra8x8Blks}} + 4 * N_{\text{intra16x16Blks}} + 16 * N_{\text{intra32x32Blks}} + 64 * N_{\text{intra64x64Blks}} \quad (6-32)$$

where $N_{\text{intra8x8Blks}}$, $N_{\text{intra16x16Blks}}$, $N_{\text{intra32x32Blks}}$ and $N_{\text{intra64x64Blks}}$ are the number of intra predicted 8x8, 16x16, 32x32 and 64x64 blocks respectively, in the specified period.

$N_{\text{intraBlks}}$ is derived from portion_intra_predicted_blocks_area and $N_{4x4BlksInPeriod}$ in the decoder.

portion_planar_blocks_in_intra_area indicates the portion of planar blocks area in the intra predicted area in the specified period and is set equal to $P_{\text{planarBlksInIntra}}$ defined as follows:

$$P_{\text{planarBlksInIntra}} = \text{Floor} \left(\frac{N_{\text{planarBlks}}}{4 * N_{\text{intraBlks}}} * 255 \right) \quad (6-33)$$

When not present, it is set equal to 0.

$N_{\text{planarBlks}}$ is the number of intra planar predicted blocks in the specified period using 4x4 granularity. At the encoder side, it is computed as follows:

$$N_{\text{planarBlks}} = N_{\text{planar4x4Blks}} + 4 * N_{\text{planar8x8Blks}} + 16 * N_{\text{planar16x16Blks}} + 64 * N_{\text{planar32x32Blks}} + 256 * N_{\text{planar64x64Blks}} \quad (6-34)$$

where $N_{\text{planar4x4Blks}}$, $N_{\text{planar8x8Blks}}$, $N_{\text{planar16x16Blks}}$, $N_{\text{planar32x32Blks}}$ and $N_{\text{planar64x64Blks}}$ are the number of intra planar predicted 4x4, 8x8, 16x16, 32x32 and 64x64 blocks respectively, in the specified period.

$N_{\text{planarBlks}}$ is derived from portion_planar_blocks_in_intra_area and $N_{\text{intraBlks}}$ in the decoder.

portion_dc_blocks_in_intra_area indicates the portion of DC blocks area in the intra predicted area in the specified period and is set equal to $P_{\text{DCBlksInIntra}}$ defined as follows:

$$P_{\text{DCBlksInIntra}} = \text{Floor} \left(\frac{N_{\text{DCBlks}}}{4 * N_{\text{intraBlks}}} * 255 \right) \quad (6-35)$$

When not present, it is set equal to 0.

N_{DCBlks} is the number of intra DC predicted blocks in the specified period using 4x4 granularity. At the encoder side, it is computed as follows:

$$N_{\text{DCBlks}} = N_{\text{DC4x4Blks}} + 4 * N_{\text{DC8x8Blks}} + 16 * N_{\text{DC16x16Blks}} + 64 * N_{\text{DC32x32Blks}} + 256 * N_{\text{DC64x64Blks}} \quad (6-36)$$

where $N_{\text{DC4x4Blks}}$, $N_{\text{DC8x8Blks}}$, $N_{\text{DC16x16Blks}}$, $N_{\text{DC32x32Blks}}$ and $N_{\text{DC64x64Blks}}$ are the number of intra DC predicted 4x4, 8x8, 16x16, 32x32 and 64x64 blocks respectively, in the specified period.

N_{DCBlks} is derived from $\text{portion_dc_blocks_in_intra_area}$ and $N_{\text{intraBlks}}$ in the decoder.

$\text{portion_angular_hv_blocks_in_intra_area}$ indicates the portion of angular horizontal or vertical blocks area in the intra predicted area in the specified period and is set equal to $P_{\text{HVBlksInIntra}}$ defined as follows:

$$P_{\text{HVBlksInIntra}} = \text{Floor} \left(\frac{N_{\text{HVBlks}}}{4 * N_{\text{intraBlks}}} * 255 \right) \quad (6-37)$$

When not present, it is set equal to 0.

$N_{\text{angularHVBlks}}$ is the number of intra angular horizontally or vertically predicted blocks in the specified period using 4x4 granularity. At the encoder side, it is computed as follows:

$$N_{\text{angularHVBlks}} = N_{\text{angularHV4x4Blks}} + 4 * N_{\text{angularHV8x8Blks}} + 16 * N_{\text{angularHV16x16Blks}} + 64 * N_{\text{angularHV32x32Blks}} + 256 * N_{\text{angularHV64x64Blks}} \quad (6-38)$$

where $N_{\text{angularHV4x4Blks}}$, $N_{\text{angularHV8x8Blks}}$, $N_{\text{angularHV16x16Blks}}$, $N_{\text{angularHV32x32Blks}}$ and $N_{\text{angularHV64x64Blks}}$ are the number of intra angular horizontally or vertically predicted 4x4, 8x8, 16x16, 32x32 and 64x64 blocks respectively, in the specified period.

$N_{\text{angularHVBlks}}$ is derived from $\text{portion_angular_hv_blocks_in_intra_area}$ and $N_{\text{intraBlks}}$ in the decoder.

$\text{portion_blocks_a_c_d_n_filterings}$ indicates the portion of prediction blocks whose luma samples position are located in sub-sample position a, c, d or n, as defined in [Annex B](#), in the specified period and is set equal to $P_{\text{acdnFiltBlks}}$ defined as follows:

$$P_{\text{acdnFiltBlks}} = \text{Floor} \left(\frac{N_{\text{acdnFiltBlks}}}{N_{\text{4x4BlksInPeriod}}} * 255 \right) \quad (6-39)$$

When not present, it is set equal to 0.

$N_{\text{acdnFiltBlks}}$ is the number of prediction blocks whose luma samples position are located in sub-sample position a, c, d or n, as defined in [Annex B](#), in the specified period. It is derived from $\text{portion_blocks_a_c_d_n_filterings}$ and $N_{\text{4x4BlksInPeriod}}$ in the decoder.

$\text{portion_blocks_h_b_filterings}$ indicates the portion of prediction blocks whose luma samples position are located in sub-sample position h or b, as defined in [Annex B](#), in the specified period and is set equal to $P_{\text{hbFiltBlks}}$ defined as follows:

$$P_{\text{hbFiltBlks}} = \text{Floor} \left(\frac{N_{\text{hbFiltBlks}}}{N_{\text{4x4BlksInPeriod}}} * 255 \right) \quad (6-40)$$

When not present, it is set equal to 0.

$N_{\text{hbFiltBlks}}$ is the number of prediction blocks whose luma samples position are located in sub-sample position h or b, as defined in [Annex B](#), in the specified period.

It is derived from $\text{portion_blocks_h_b_filterings}$ and $N_{\text{4x4BlksInPeriod}}$ in the decoder.

$\text{portion_blocks_f_i_k_q_filterings}$ indicates the portion of prediction blocks whose luma samples position are located in sub-sample position f, i, k or q, as defined in [Annex B](#), in the specified period and is set equal to $P_{\text{fikqFiltBlks}}$ defined as follows:

$$P_{\text{fikqFilt}} = \text{Floor} \left(\frac{N_{\text{fikqFiltBlks}}}{N_{\text{4x4BlksInPeriod}}} * 255 \right) \quad (6-41)$$

When not present, it is set equal to 0.

$N_{fikqFiltBlks}$ is the number of prediction blocks whose luma samples position are located in sub-sample position f, i, k or q as defined in [Annex B](#), in the specified period.

It is derived from $portion_blocks_f_i_k_q_filterings$ and $N_{4x4BlksInPeriod}$ in the decoder.

$portion_blocks_j_filterings$ indicates the portion of prediction blocks whose luma samples position are located in sub-sample position j , as defined in [Annex B](#), in the specified period and is set equal to $P_{jFiltBlks}$ defined as follows:

$$P_{jFiltBlks} = \text{Floor} \left(\frac{N_{jFiltBlks}}{N_{4x4BlksInPeriod}} * 255 \right) \quad (6-42)$$

When not present, it is set equal to 0.

$N_{jFiltBlks}$ is the number of prediction blocks whose luma samples position are located in sub-sample position j , as defined in [Annex B](#), in the specified period.

It is derived from $portion_blocks_j_filterings$ and $N_{4x4BlksInPeriod}$ in the decoder.

$portion_blocks_e_g_p_r_filterings$ indicates the portion of prediction blocks whose luma blocks position are located in sub-sample position e, g, p or r , as defined in [Annex B](#), in the specified period and is set equal to $P_{egprFiltBlks}$ defined as follows:

$$P_{egprFiltBlks} = \text{Floor} \left(\frac{N_{egprFiltBlks}}{N_{4x4BlksInPeriod}} * 255 \right) \quad (6-43)$$

When not present, it is set equal to 0.

$N_{egprFiltBlks}$ is the number of prediction blocks whose luma samples position are located in sub-sample position e, g, p or r , as defined in [Annex B](#), in the specified period.

It is derived from $portion_blocks_e_g_p_r_filterings$ and $N_{4x4BlksInPeriod}$ in the decoder.

$portion_deblocking_instances$ indicates the portion of deblocking filtering instances in the specified period and is set equal to $P_{dbfInstances}$ defined as follows:

$$P_{dbfInstances} = \text{Floor} \left(\frac{N_{dbfInstances}}{4 * S_{chrMult} * N_{4x4BlksInPeriod}} * 255 \right) \quad (6-44)$$

$S_{chrMult}$ depends on the HEVC variables $separate_colour_plane_flag$ and $chroma_format_idc$ as shown in the [Table 7](#).

Table 7 — specification of $S_{chrMult}$ for HEVC

$S_{chrMult}$	$separate_colour_plane_flag$	$chroma_format_idc$	Comment
1	0	0	monochrome
1.5	0	1	4:2:0 sampling
2	0	2	4:2:2 sampling
3	0	3	4:4:4 sampling
3	1	3	separate colour plane

$N_{dbfInstances}$ is the number of deblocking filtering instances in the specified period. It is derived from $portion_deblocking_instances$, $N_{4x4BlksInPeriod}$ and $S_{chrMult}$ in the decoder.

$max_num_slices_tiles_minus1$ specifies the maximum number between the number of slices and the number of tiles in the associated picture.