Table 9-42 – Specification of ctxBlockCat for the different blocks

Block description	maxNumCoeff	ctxBlockCat
block of luma DC transform coefficient levels (i.e., list Intra16x16DCLevel as described in subclause 7.4.5.3)	16	0
block of luma AC transform coefficient levels (i.e., list Intra16x16ACLevel[i] as described in subclause 7.4.5.3)	15	1
block of 16 luma transform coefficient levels (i.e., list LumaLevel[i] as described in subclause 7.4.5.3)	16	2
block of chroma DC transform coefficient levels when ChromaArrayType is equal to 1 or 2 (i.e., list ChromaDCLevel as described in subclause 7.4.5.3)	4 * NumC8x8	3
block of chroma AC transform coefficient levels when ChromaArrayType is equal to 1 or 2 (i.e., list ChromaACLevel as described in subclause 7.4.5.3)	15	4
block of 64 luma transform coefficient levels (i.e., list LumaLevel8x8[i] as described in subclause 7.4.5.3)	64	5
block of Cb DC transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CbIntra16x16DCLevel as described in subclause 7.4.5.3)	16	6
block of Cb AC transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CbIntra16x16ACLevel[i] as described in subclause 7.4.5.3)	15	7
block of 16 Cb transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CbLevel[i] as described in subclause 7.4.5.3)	16	8
block of 64 Cb transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CbLevel8x8[i] as described in subclause 7.4.5.3)	64	9
block of Cr DC transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CrIntra16x16DCLevel as described in subclause 7.4.5.3)	16	10
block of Cr AC transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CrIntra16x16ACLevel[i] as described in subclause 7.4.5.3)	15	11
block of 16 Cr transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CrLevel[i] as described in subclause 7.4.5.3)	16	12
block of 64 Cr transform coefficient levels when ChromaArrayType is equal to 3 (i.e., list CrLevel8x8[i] as described in subclause 7.4.5.3)	64	13

Let the variable levelListIdx be set equal to the index of the list of transform coefficient levels as specified in subclause 7.4.5.3.

For the syntax elements significant_coeff_flag and last_significant_coeff_flag in blocks with ctxBlockCat not equal to 3, 5, 9, and 13, the variable ctxIdxInc is derived by

$$ctxIdxInc = levelListIdx \tag{9-15}$$

where levelListIdx ranges from 0 to maxNumCoeff – 2, inclusive.

For the syntax elements significant_coeff_flag and last_significant_coeff_flag in blocks with ctxBlockCat == 3, the variable ctxIdxInc is derived by

$$ctxIdxInc = Min(levelListIdx / NumC8x8, 2) \tag{9-16}$$

where levelListIdx ranges from 0 to 4 * NumC8x8 – 2, inclusive.

For the syntax elements significant_coeff_flag and last_significant_coeff_flag in 8x8 luma, Cb, or Cr blocks with ctxBlockCat == 5, 9, or 13, Table 9-43 contains the specification of ctxIdxInc for the given values of levelListIdx, where levelListIdx ranges from 0 to 62, inclusive.

Google
Exhibit 1032

Table 9-43 – Mapping of scanning position to ctxIdxInc for ctxBlockCat == 5, 9, or 13

levelListIdx	ctxIdxInc for significant_coeff_flag (frame coded macroblocks)	ctxIdxInc for significant_coeff_flag (field coded macroblocks)	ctxIdxInc for last_significant_coeff_flag	levelListIdx	ctxIdxInc for significant_coeff_flag (frame coded macroblocks)	ctxIdxInc for significant_coeff_flag (field coded macroblocks)	ctxIdxInc for last_significant_coeff_flag
0	0	0	0	**32**	7	9	3
1	1	1	1	**33**	6	9	3
2	2	1	1	**34**	11	10	3
3	3	2	1	**35**	12	10	3
4	4	2	1	**36**	13	8	3
5	5	3	1	**37**	11	11	3
6	5	3	1	**38**	6	12	3
7	4	4	1	**39**	7	11	3
8	4	5	1	**40**	8	9	4
9	3	6	1	**41**	9	9	4
10	3	7	1	**42**	14	10	4
11	4	7	1	**43**	10	10	4
12	4	7	1	**44**	9	8	4
13	4	8	1	**45**	8	13	4
14	5	4	1	**46**	6	13	4
15	5	5	1	**47**	11	9	4
16	4	6	2	**48**	12	9	5
17	4	9	2	**49**	13	10	5
18	4	10	2	**50**	11	10	5
19	4	10	2	**51**	6	8	5
20	3	8	2	**52**	9	13	6
21	3	11	2	**53**	14	13	6
22	6	12	2	**54**	10	9	6
23	7	11	2	**55**	9	9	6
24	7	9	2	**56**	11	10	7
25	7	9	2	**57**	12	10	7
26	8	10	2	**58**	13	14	7
27	9	10	2	**59**	11	14	7

Table 9-43 – Mapping of scanning position to ctxIdxInc for ctxBlockCat == 5, 9, or 13

levelListIdx	ctxIdxInc for significant_coeff_flag (frame coded macroblocks)	ctxIdxInc for significant_coeff_flag (field coded macroblocks)	ctxIdxInc for last_significant_coeff_flag	levelListIdx	ctxIdxInc for significant_coeff_flag (frame coded macroblocks)	ctxIdxInc for significant_coeff_flag (field coded macroblocks)	ctxIdxInc for last_significant_coeff_flag
28	10	8	2	**60**	14	14	8
29	9	11	2	**61**	10	14	8
30	8	12	2	**62**	12	14	8
31	7	11	2				

Let numDecodAbsLevelEq1 denotes the accumulated number of decoded transform coefficient levels with absolute value equal to 1, and let numDecodAbsLevelGt1 denotes the accumulated number of decoded transform coefficient levels with absolute value greater than 1. Both numbers are related to the same transform coefficient block, where the current decoding process takes place. Then, for decoding of coeff_abs_level_minus1, ctxIdxInc for coeff_abs_level_minus1 is specified depending on binIdx as follows.

– If binIdx is equal to 0, ctxIdxInc is derived by

$$ctxIdxInc = ((numDecodAbsLevelGt1 \; != \; 0) \; ? \; 0 : Min(4, 1 + numDecodAbsLevelEq1)) \tag{9-17}$$

– Otherwise (binIdx is greater than 0), ctxIdxInc is derived by

$$ctxIdxInc = 5 + Min(4 - (ctxBlockCat \; == \; 3), numDecodAbsLevelGt1) \tag{9-18}$$

9.3.3.2 Arithmetic decoding process

Inputs to this process are the bypassFlag, ctxIdx as derived in subclause 9.3.3.1, and the state variables codIRange and codIOffset of the arithmetic decoding engine.

Output of this process is the value of the bin.

Figure 9-2 illustrates the whole arithmetic decoding process for a single bin. For decoding the value of a bin, the context index ctxIdx is passed to the arithmetic decoding process DecodeBin(ctxIdx), which is specified as follows.

– If bypassFlag is equal to 1, DecodeBypass( ) as specified in subclause 9.3.3.2.3 is invoked.

– Otherwise, if bypassFlag is equal to 0 and ctxIdx is equal to 276, DecodeTerminate( ) as specified in subclause 9.3.3.2.4 is invoked.

– Otherwise (bypassFlag is equal to 0 and ctxIdx is not equal to 276), DecodeDecision( ) as specified in subclause 9.3.3.2.1 is applied.

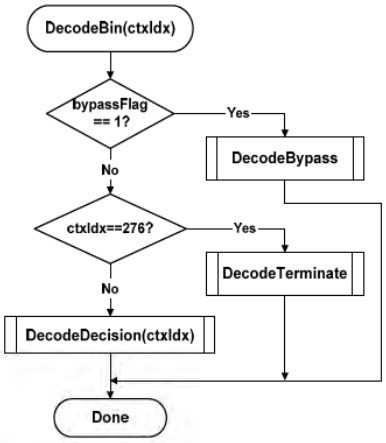


Figure 9-2 – Overview of the arithmetic decoding process for a single bin (informative)

NOTE – Arithmetic coding is based on the principle of recursive interval subdivision. Given a probability estimation p(0) and p(1) = 1 – p(0) of a binary decision (0, 1), an initially given code sub-interval with the range codIRange will be subdivided into two sub-intervals having range p(0) * codIRange and codIRange – p(0) * codIRange, respectively. Depending on the decision, which has been observed, the corresponding sub-interval will be chosen as the new code interval, and a binary code string pointing into that interval will represent the sequence of observed binary decisions. It is useful to distinguish between the most probable symbol (MPS) and the least probable symbol (LPS), so that binary decisions have to be identified as either MPS or LPS, rather than 0 or 1. Given this terminology, each context is specified by the probability p_{LPS} of the LPS and the value of MPS (valMPS), which is either 0 or 1.

The arithmetic core engine in this Recommendation | International Standard has three distinct properties:

- The probability estimation is performed by means of a finite-state machine with a table-based transition process between 64 different representative probability states { p_{LPS}(pStateIdx) | 0 <= pStateIdx < 64 } for the LPS probability p_{LPS}. The numbering of the states is arranged in such a way that the probability state with index pStateIdx = 0 corresponds to an LPS probability value of 0.5, with decreasing LPS probability towards higher state indices.
- The range codIRange representing the state of the coding engine is quantised to a small set $\{Q_1,\ldots,Q_4\}$ of pre-set quantisation values prior to the calculation of the new interval range. Storing a table containing all 64x4 pre-computed product values of $Q_i * p_{LPS}$(pStateIdx) allows a multiplication-free approximation of the product codIRange * p_{LPS}(pStateIdx).
- For syntax elements or parts thereof for which an approximately uniform probability distribution is assumed to be given a separate simplified encoding and decoding bypass process is used.

9.3.3.2.1 Arithmetic decoding process for a binary decision

Inputs to this process are ctxIdx, codIRange, and codIOffset.

Outputs of this process are the decoded value binVal, and the updated variables codIRange and codIOffset.

Figure 9-3 shows the flowchart for decoding a single decision (DecodeDecision).

1. The value of the variable codIRangeLPS is derived as follows.

– Given the current value of codIRange, the variable qCodIRangeIdx is derived by

$$qCodIRangeIdx = (codIRange >> 6) \& 0x03 \quad (9\text{-}19)$$

– Given qCodIRangeIdx and pStateIdx associated with ctxIdx, the value of the variable rangeTabLPS as specified in Table 9-44 is assigned to codIRangeLPS:

$$codIRangeLPS = rangeTabLPS[pStateIdx][qCodIRangeIdx] \quad (9\text{-}20)$$

2. The variable codIRange is set equal to codIRange – codIRangeLPS and the following applies.

– If codIOffset is greater than or equal to codIRange, the variable binVal is set equal to 1 - valMPS, codIOffset is decremented by codIRange, and codIRange is set equal to codIRangeLPS.

– Otherwise, the variable binVal is set equal to valMPS.

Given the value of binVal, the state transition is performed as specified in subclause 9.3.3.2.1.1. Depending on the current value of codIRange, renormalization is performed as specified in subclause 9.3.3.2.2.

9.3.3.2.1.1 State transition process

Inputs to this process are the current pStateIdx, the decoded value binVal and valMPS values of the context variable associated with ctxIdx.

Outputs of this process are the updated pStateIdx and valMPS of the context variable associated with ctxIdx.

Depending on the decoded value binVal, the update of the two variables pStateIdx and valMPS associated with ctxIdx is derived as follows:

```
if( binVal == valMPS )
    pStateIdx = transIdxMPS( pStateIdx )
else {                                                    (9-21)
    if( pStateIdx == 0 )
        valMPS = 1 – valMPS
    pStateIdx = transIdxLPS( pStateIdx )
}
```

Table 9-45 specifies the transition rules transIdxMPS( ) and transIdxLPS( ) after decoding the value of valMPS and 1 – valMPS, respectively.

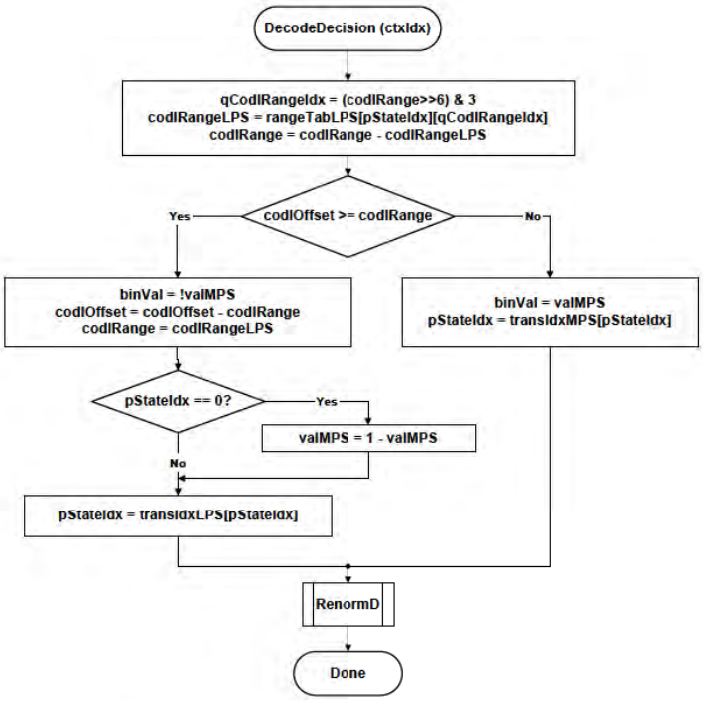


Figure 9-3 – Flowchart for decoding a decision

Table 9-44 – Specification of rangeTabLPS depending on pStateIdx and qCodIRangeIdx

pStateIdx	qCodIRangeIdx				pStateIdx	qCodIRangeIdx			
	0	1	2	3		0	1	2	3
0	128	176	208	240	32	27	33	39	45
1	128	167	197	227	33	26	31	37	43
2	128	158	187	216	34	24	30	35	41
3	123	150	178	205	35	23	28	33	39
4	116	142	169	195	36	22	27	32	37
5	111	135	160	185	37	21	26	30	35
6	105	128	152	175	38	20	24	29	33
7	100	122	144	166	39	19	23	27	31
8	95	116	137	158	40	18	22	26	30
9	90	110	130	150	41	17	21	25	28
10	85	104	123	142	42	16	20	23	27
11	81	99	117	135	43	15	19	22	25
12	77	94	111	128	44	14	18	21	24
13	73	89	105	122	45	14	17	20	23
14	69	85	100	116	46	13	16	19	22
15	66	80	95	110	47	12	15	18	21
16	62	76	90	104	48	12	14	17	20
17	59	72	86	99	49	11	14	16	19
18	56	69	81	94	50	11	13	15	18
19	53	65	77	89	51	10	12	15	17
20	51	62	73	85	52	10	12	14	16
21	48	59	69	80	53	9	11	13	15
22	46	56	66	76	54	9	11	12	14
23	43	53	63	72	55	8	10	12	14
24	41	50	59	69	56	8	9	11	13
25	39	48	56	65	57	7	9	11	12
26	37	45	54	62	58	7	9	10	12
27	35	43	51	59	59	7	8	10	11
28	33	41	48	56	60	6	8	9	11
29	32	39	46	53	61	6	7	9	10
30	30	37	43	50	62	6	7	8	9
31	29	35	41	48	63	2	2	2	2

Table 9-45 – State transition table

pStateIdx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
transIdxLPS	0	0	1	2	2	4	4	5	6	7	8	9	9	11	11	12
transIdxMPS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
pStateIdx	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
transIdxLPS	13	13	15	15	16	16	18	18	19	19	21	21	22	22	23	24
transIdxMPS	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
pStateIdx	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
transIdxLPS	24	25	26	26	27	27	28	29	29	30	30	30	31	32	32	33
transIdxMPS	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
pStateIdx	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
transIdxLPS	33	33	34	34	35	35	35	36	36	36	37	37	37	38	38	63
transIdxMPS	49	50	51	52	53	54	55	56	57	58	59	60	61	62	62	63

9.3.3.2.2 Renormalization process in the arithmetic decoding engine

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

Outputs of this process are the updated variables codIRange and codIOffset.

A flowchart of the renormalization is shown in Figure 9-4. The current value of codIRange is first compared to 0x0100 and further steps are specified as follows.

– If codIRange is greater than or equal to 0x0100, no renormalization is needed and the RenormD process is finished;

– Otherwise (codIRange is less than 0x0100), the renormalization loop is entered. Within this loop, the value of codIRange is doubled, i.e., left-shifted by 1 and a single bit is shifted into codIOffset by using read_bits(1).

The bitstream shall not contain data that results in a value of codIOffset being greater than or equal to codIRange upon completion of this process.

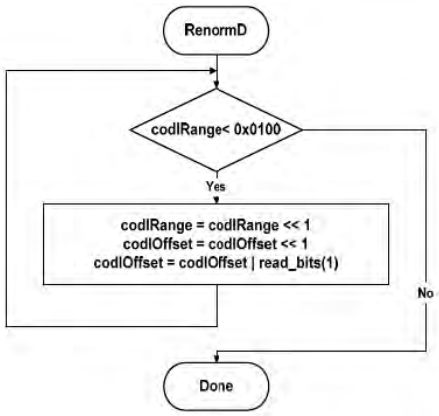


Figure 9-4 – Flowchart of renormalization

9.3.3.2.3 Bypass decoding process for binary decisions

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

Outputs of this process are the updated variable codIOffset and the decoded value binVal.

The bypass decoding process is invoked when bypassFlag is equal to 1. Figure 9-5 shows a flowchart of the corresponding process.

First, the value of codIOffset is doubled, i.e., left-shifted by 1 and a single bit is shifted into codIOffset by using read_bits(1). Then, the value of codIOffset is compared to the value of codIRange and further steps are specified as follows.

– If codIOffset is greater than or equal to codIRange, the variable binVal is set equal to 1 and codIOffset is decremented by codIRange.

– Otherwise (codIOffset is less than codIRange), the variable binVal is set equal to 0.

The bitstream shall not contain data that results in a value of codIOffset being greater than or equal to codIRange upon completion of this process.

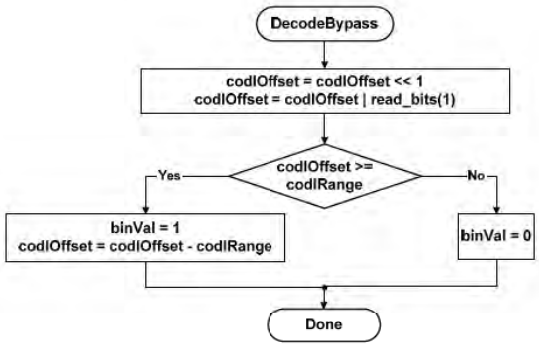


Figure 9-5 – Flowchart of bypass decoding process

9.3.3.2.4 Decoding process for binary decisions before termination

Inputs to this process are bits from slice data and the variables codIRange and codIOffset.

Outputs of this process are the updated variables codIRange and codIOffset, and the decoded value binVal.

This special decoding routine applies to decoding of end_of_slice_flag and of the bin indicating the I_PCM mode corresponding to ctxIdx equal to 276. Figure 9-6 shows the flowchart of the corresponding decoding process, which is specified as follows.

First, the value of codIRange is decremented by 2. Then, the value of codIOffset is compared to the value of codIRange and further steps are specified as follows.

– If codIOffset is greater than or equal to codIRange, the variable binVal is set equal to 1, no renormalization is carried out, and CABAC decoding is terminated. The last bit inserted in register codIOffset is equal to 1. When decoding end_of_slice_flag, this last bit inserted in register codIOffset is interpreted as rbsp_stop_one_bit.

– Otherwise (codIOffset is less than codIRange), the variable binVal is set equal to 0 and renormalization is performed as specified in subclause 9.3.3.2.2.

NOTE – This procedure may also be implemented using DecodeDecision(ctxIdx) with ctxIdx = 276. In the case where the decoded value is equal to 1, seven more bits would be read by DecodeDecision(ctxIdx) and a decoding process would have to adjust its bitstream pointer accordingly to properly decode following syntax elements.

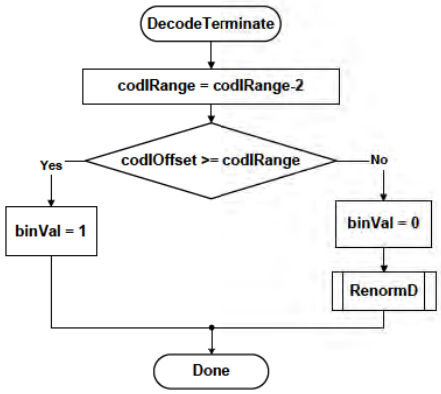


Figure 9-6 – Flowchart of decoding a decision before termination

9.3.4 Arithmetic encoding process (informative)

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

Inputs to this process are decisions that are to be encoded and written.

Outputs of this process are bits that are written to the RBSP.

This informative subclause describes an arithmetic encoding engine that matches the arithmetic decoding engine described in subclause 9.3.3.2. The encoding engine is essentially symmetric with the decoding engine, i.e., procedures are called in the same order. The following procedures are described in this section: InitEncoder, EncodeDecision, EncodeBypass, EncodeTerminate, which correspond to InitDecoder, DecodeDecision, DecodeBypass, and DecodeTerminate, respectively. The state of the arithmetic encoding engine is represented by a value of the variable codILow pointing to the lower end of a sub-interval and a value of the variable codIRange specifying the corresponding range of that sub-interval.

9.3.4.1 Initialisation process for the arithmetic encoding engine (informative)

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

This process is invoked before encoding the first macroblock of a slice, and after encoding any pcm_alignment_zero_bit and all pcm_sample_luma and pcm_sample_chroma data for a macroblock of type I_PCM.

Outputs of this process are the values codILow, codIRange, firstBitFlag, bitsOutstanding, and symCnt of the arithmetic encoding engine.

In the initialisation procedure of the encoder, codILow is set equal to 0, and codIRange is set equal to 0x01FE. Furthermore, a firstBitFlag is set equal to 1, and bitsOutstanding and symCnt counters are set equal to 0.

NOTE – The minimum register precision required for codILow is 10 bits and for CodIRange is 9 bits. The precision required for the counters bitsOutstanding and symCnt should be sufficiently large to prevent overflow of the related registers. When MaxBinCountInSlice denotes the maximum total number of binary decisions to encode in one slice, the minimum register precision required for the variables bitsOutstanding and symCnt is given by Ceil(Log2(MaxBinCountInSlice + 1)).

9.3.4.2 Encoding process for a binary decision (informative)

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

Inputs to this process are the context index ctxIdx, the value of binVal to be encoded, and the variables codIRange, codILow and symCnt.

Outputs of this process are the variables codIRange, codILow, and symCnt.

Figure 9-7 shows the flowchart for encoding a single decision. In a first step, the variable codIRangeLPS is derived as follows.

Given the current value of codIRange, codIRange is mapped to the index qCodIRangeIdx of a quantised value of codIRange by using Equation 9-19. The value of qCodIRangeIdx and the value of pStateIdx associated with ctxIdx are used to determine the value of the variable rangeTabLPS as specified in Table 9-44, which is assigned to codIRangeLPS. The value of codIRange – codIRangeLPS is assigned to codIRange.

In a second step, the value of binVal is compared to valMPS associated with ctxIdx. When binVal is different from valMPS, codIRange is added to codILow and codIRange is set equal to the value codIRangeLPS. Given the encoded decision, the state transition is performed as specified in subclause 9.3.3.2.1.1. Depending on the current value of codIRange, renormalization is performed as specified in subclause 9.3.4.3. Finally, the variable symCnt is incremented by 1.

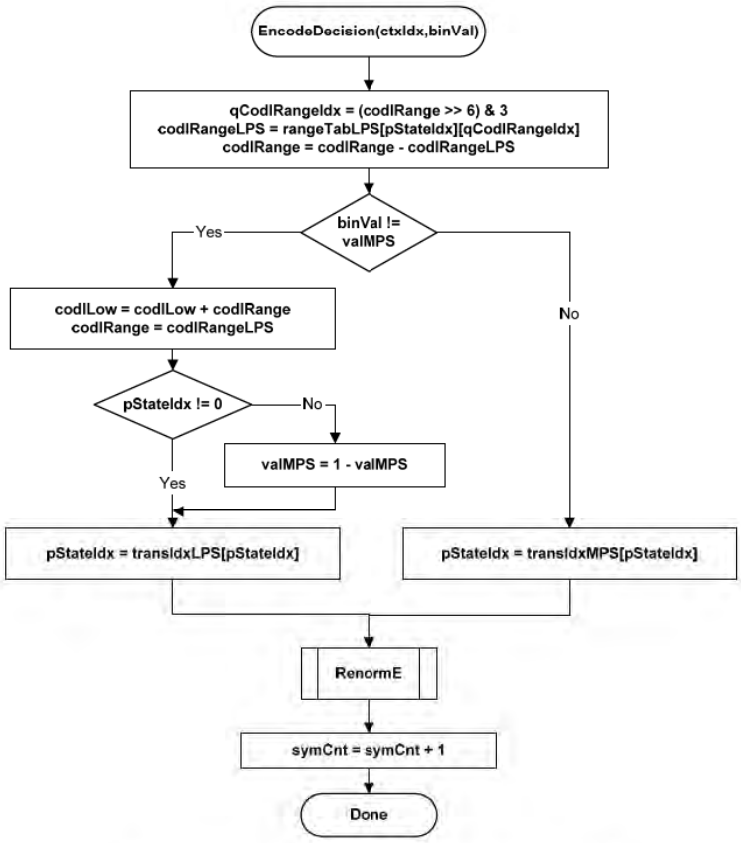


Figure 9-7 – Flowchart for encoding a decision

9.3.4.3 Renormalization process in the arithmetic encoding engine (informative)

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

Inputs to this process are the variables codIRange, codILow, firstBitFlag, and bitsOutstanding.

Outputs of this process are zero or more bits written to the RBSP and the updated variables codIRange, codILow, firstBitFlag, and bitsOutstanding.

Renormalization is illustrated in Figure 9-8.

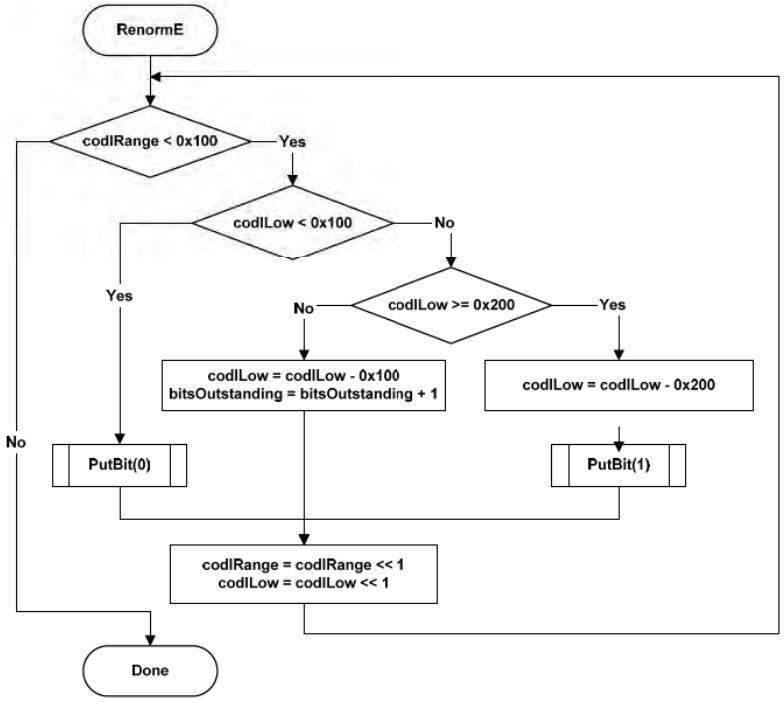


Figure 9-8 – Flowchart of renormalization in the encoder

The PutBit( ) procedure described in Figure 9-9 provides carry over control. It uses the function WriteBits(B, N) that writes N bits with value B to the bitstream and advances the bitstream pointer by N bit positions. This function assumes the existence of a bitstream pointer with an indication of the position of the next bit to be written to the bitstream by the encoding process.

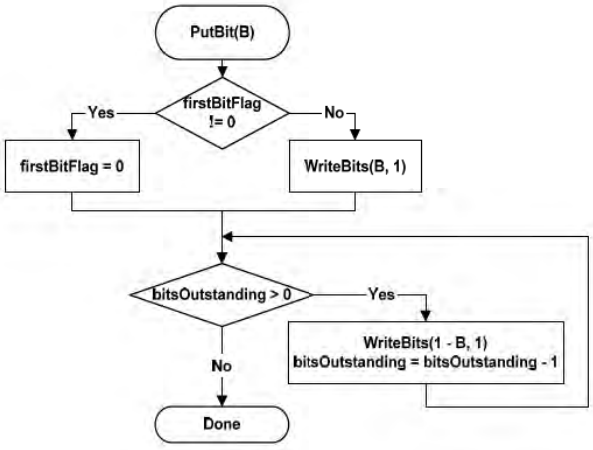


Figure 9-9 – Flowchart of PutBit(B)

9.3.4.4 Bypass encoding process for binary decisions (informative)

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

Inputs to this process are the variables binVal, codILow, codIRange, bitsOutstanding, and symCnt.

Output of this process is a bit written to the RBSP and the updated variables codILow, bitsOutstanding, and symCnt.

This encoding process applies to all binary decisions with bypassFlag equal to 1. Renormalization is included in the specification of this process as given in Figure 9-10.

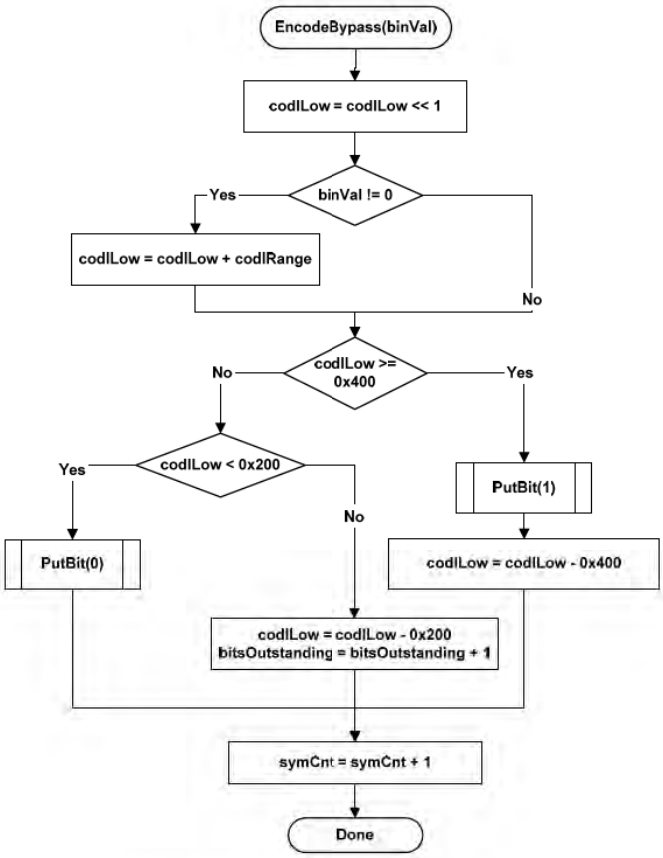


Figure 9-10 – Flowchart of encoding bypass

9.3.4.5 Encoding process for a binary decision before termination (informative)

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

Inputs to this process are the variables binVal, codIRange, codILow, bitsOutstanding, and symCnt.

Outputs of this process are zero or more bits written to the RBSP and the updated variables codILow, codIRange, bitsOutstanding, and symCnt.

This encoding routine shown in Figure 9-11 applies to encoding of the end_of_slice_flag and of the bin indicating the I_PCM mb_type both associated with ctxIdx equal to 276.

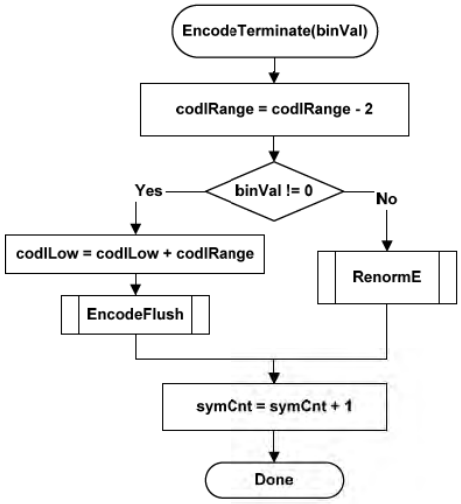


Figure 9-11 – Flowchart of encoding a decision before termination

When the value of binVal to encode is equal to 1, CABAC encoding is terminated and the flushing procedure shown in Figure 9-12 is applied. In this flushing procedure, the last bit written by WriteBits(B, N) is equal to 1. When encoding end_of_slice_flag, this last bit is interpreted as the rbsp_stop_one_bit.

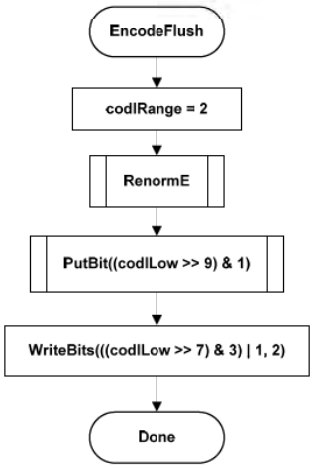


Figure 9-12 – Flowchart of flushing at termination

9.3.4.6 Byte stuffing process (informative)

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

This process is invoked after encoding the last macroblock of the last slice of a picture and after encapsulation.

Inputs to this process are the number of bytes NumBytesInVclNALunits of all VCL NAL units of a picture, the number of macroblocks PicSizeInMbs in the picture, and the number of binary symbols BinCountsInNALunits resulting from encoding the contents of all VCL NAL units of the picture.

Outputs of this process are zero or more bytes appended to the NAL unit.

Let the variable k be set equal to Ceil((Ceil(3 * (32 * BinCountsInNALunits – RawMbBits * PicSizeInMbs) ÷ 1024) – NumBytesInVclNALunits) ÷ 3). Depending on the variable k the following applies.

– If k is less than or equal to 0, no cabac_zero_word is appended to the NAL unit.

– Otherwise (k is greater than 0), the 3-byte sequence 0x000003 is appended k times to the NAL unit after encapsulation, where the first two bytes 0x0000 represent a cabac_zero_word and the third byte 0x03 represents an emulation_prevention_three_byte.

Annex A
Profiles and levels

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

Profiles and levels specify restrictions on bitstreams and hence limits on the capabilities needed to decode the bitstreams. Profiles and levels may also be used to indicate interoperability points between individual decoder implementations.

NOTE 1 – This Recommendation | International Standard does not include individually selectable "options" at the decoder, as this would increase interoperability difficulties.

Each profile specifies a subset of algorithmic features and limits that shall be supported by all decoders conforming to that profile.

NOTE 2 – Encoders are not required to make use of any particular subset of features supported in a profile.

Each level specifies a set of limits on the values that may be taken by the syntax elements of this Recommendation | International Standard. The same set of level definitions is used with all profiles, but individual implementations may support a different level for each supported profile. For any given profile, levels generally correspond to decoder processing load and memory capability.

A.1 Requirements on video decoder capability

Capabilities of video decoders conforming to this Recommendation | International Standard are specified in terms of the ability to decode video streams conforming to the constraints of profiles and levels specified in this annex. For each such profile, the level supported for that profile shall also be expressed.

Specific values are specified in this annex for the syntax elements profile_idc and level_idc. All other values of profile_idc and level_idc are reserved for future use by ITU-T | ISO/IEC.

NOTE – Decoders should not infer that when a reserved value of profile_idc or level_idc falls between the values specified in this Recommendation | International Standard that this indicates intermediate capabilities between the specified profiles or levels, as there are no restrictions on the method to be chosen by ITU-T | ISO/IEC for the use of such future reserved values.

A.2 Profiles

A.2.1 Baseline profile

Bitstreams conforming to the Baseline profile shall obey the following constraints:

- Only I and P slice types may be present.
- NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
- Sequence parameter sets shall have frame_mbs_only_flag equal to 1.
- The syntax elements chroma_format_idc, bit_depth_luma_minus8, bit_depth_chroma_minus8, qpprime_y_zero_transform_bypass_flag, and seq_scaling_matrix_present_flag shall not be present in sequence parameter sets.
- Picture parameter sets shall have weighted_pred_flag and weighted_bipred_idc both equal to 0.
- Picture parameter sets shall have entropy_coding_mode_flag equal to 0.
- Picture parameter sets shall have num_slice_groups_minus1 in the range of 0 to 7, inclusive.
- The syntax elements transform_8x8_mode_flag, pic_scaling_matrix_present_flag, and second_chroma_qp_index_offset shall not be present in picture parameter sets.
- The syntax element level_prefix shall not be greater than 15.
- The level constraints specified for the Baseline profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Baseline profile is specified by profile_idc being equal to 66.

Decoders conforming to the Baseline profile at a specific level shall be capable of decoding all bitstreams in which profile_idc is equal to 66 or constraint_set0_flag is equal to 1 and in which level_idc and constraint_set3_flag represent a level less than or equal to the specified level.

A.2.2 Main profile

Bitstreams conforming to the Main profile shall obey the following constraints:

- Only I, P, and B slice types may be present.
- NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
- Arbitrary slice order is not allowed.
- The syntax elements chroma_format_idc, bit_depth_luma_minus8, bit_depth_chroma_minus8, qpprime_y_zero_transform_bypass_flag, and seq_scaling_matrix_present_flag shall not be present in sequence parameter sets.
- Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
- Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
- The syntax elements transform_8x8_mode_flag, pic_scaling_matrix_present_flag, and second_chroma_qp_index_offset shall not be present in picture parameter sets.
- The syntax element level_prefix shall not be greater than 15 (when present).
- The level constraints specified for the Main profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Main profile is specified by profile_idc being equal to 77.

Decoders conforming to the Main profile at a specified level shall be capable of decoding all bitstreams in which profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and in which level_idc and constraint_set3_flag represent a level less than or equal to the specified level.

A.2.3 Extended profile

Bitstreams conforming to the Extended profile shall obey the following constraints:

- Sequence parameter sets shall have direct_8x8_inference_flag equal to 1.
- The syntax elements chroma_format_idc, bit_depth_luma_minus8, bit_depth_chroma_minus8, qpprime_y_zero_transform_bypass_flag, and seq_scaling_matrix_present_flag shall not be present in sequence parameter sets.
- Picture parameter sets shall have entropy_coding_mode_flag equal to 0.
- Picture parameter sets shall have num_slice_groups_minus1 in the range of 0 to 7, inclusive.
- The syntax elements transform_8x8_mode_flag, pic_scaling_matrix_present_flag, and second_chroma_qp_index_offset shall not be present in picture parameter sets.
- The syntax element level_prefix shall not be greater than 15 (when present).
- The level constraints specified for the Extended profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the Extended profile is specified by profile_idc being equal to 88.

Decoders conforming to the Extended profile at a specified level shall be capable of decoding all bitstreams in which profile_idc is equal to 88 or constraint_set2_flag is equal to 1 and in which level_idc represents a level less than or equal to specified level.

Decoders conforming to the Extended profile at a specified level shall also be capable of decoding all bitstreams in which profile_idc is equal to 66 or constraint_set0_flag is equal to 1, in which level_idc and constraint_set3_flag represent a level less than or equal to the specified level.

A.2.4 High profile

Bitstreams conforming to the High profile shall obey the following constraints:

- Only I, P, and B slice types may be present.
- NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
- Arbitrary slice order is not allowed.
- Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
- Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
- Sequence parameter sets shall have chroma_format_idc in the range of 0 to 1 inclusive.
- Sequence parameter sets shall have bit_depth_luma_minus8 equal to 0 only.
- Sequence parameter sets shall have bit_depth_chroma_minus8 equal to 0 only.

– Sequence parameter sets shall have qpprime_y_zero_transform_bypass_flag equal to 0 only.
– The level constraints specified for the High profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High profile is specified by profile_idc being equal to 100. Decoders conforming to the High profile at a specific level shall be capable of decoding all bitstreams in which either or both of the following conditions are true:

– profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and the combination of level_idc and constraint_set3_flag represent a level less than or equal to the specified level.
– profile_idc is equal to 100 and level_idc represents a level less than or equal to the specified level.

NOTE – The value 100 for profile_idc indicates that the bitstream conforms to the High profile as specified in this subclause. When profile_idc is equal to 100 and constraint_set3_flag is equal to 1, this indicates that the bitstream conforms to the High profile and additionally conforms to the constraints specified for the High 10 Intra profile in subclause A.2.8. For example, such a bitstream must have bit_depth_luma_minus8 equal to 0, have bit_depth_chroma_minus8 equal to 0, obey the MinCr, MaxBR and MaxCPB constraints of the High profile, contain only IDR pictures, have num_ref_frames equal to 0, have dpb_output_delay equal to 0, and obey the maximum slice size constraint of the High 10 Intra profile.

A.2.5 High 10 profile

Bitstreams conforming to the High 10 profile shall obey the following constraints:

– Only I, P, and B slice types may be present.
– NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
– Arbitrary slice order is not allowed.
– Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
– Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
– Sequence parameter sets shall have chroma_format_idc in the range of 0 to 1 inclusive.
– Sequence parameter sets shall have bit_depth_luma_minus8 in the range of 0 to 2 inclusive.
– Sequence parameter sets shall have bit_depth_chroma_minus8 in the range of 0 to 2 inclusive.
– Sequence parameter sets shall have qpprime_y_zero_transform_bypass_flag equal to 0 only.
– The level constraints specified for the High 10 profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 10 profile is specified by profile_idc being equal to 110. Decoders conforming to the High 10 profile at a specific level shall be capable of decoding all bitstreams in which either or both of the following conditions are true:

– profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and the combination of level_idc and constraint_set3_flag represent a level less than or equal to the specified level.
– profile_idc is equal to 100 or 110 and level_idc represents a level less than or equal to the specified level.

A.2.6 High 4:2:2 profile

Bitstreams conforming to the High 4:2:2 profile shall obey the following constraints:

– Only I, P, and B slice types may be present.
– NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
– Arbitrary slice order is not allowed.
– Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
– Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
– Sequence parameter sets shall have chroma_format_idc in the range of 0 to 2 inclusive.
– Sequence parameter sets shall have bit_depth_luma_minus8 in the range of 0 to 2 inclusive.
– Sequence parameter sets shall have bit_depth_chroma_minus8 in the range of 0 to 2 inclusive.
– Sequence parameter sets shall have qpprime_y_zero_transform_bypass_flag equal to 0 only.
– The level constraints specified for the High 4:2:2 profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 4:2:2 profile is specified by profile_idc being equal to 122. Decoders conforming to the High 4:2:2 profile at a specific level shall be capable of decoding all bitstreams in which either or both of the following conditions are true:

– profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and the combination of level_idc and constraint_set3_flag represent a level less than or equal to the specified level.

– profile_idc is equal to 100, 110, or 122 and level_idc represents a level less than or equal to the specified level.

A.2.7 High 4:4:4 Predictive profile

Bitstreams conforming to the High 4:4:4 Predictive profile shall obey the following constraints:

– Only I, P, B slice types may be present.
– NAL unit streams shall not contain nal_unit_type values in the range of 2 to 4, inclusive.
– Arbitrary slice order is not allowed.
– Picture parameter sets shall have num_slice_groups_minus1 equal to 0 only.
– Picture parameter sets shall have redundant_pic_cnt_present_flag equal to 0 only.
– Sequence parameter sets shall have bit_depth_luma_minus8 in the range of 0 to 6 inclusive.
– Sequence parameter sets shall have bit_depth_chroma_minus8 in the range of 0 to 6 inclusive.
– The level constraints specified for the High 4:4:4 Predictive profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 4:4:4 Predictive profile is specified by profile_idc being equal to 244. Decoders conforming to the High 4:4:4 Predictive profile at a specific level shall be capable of decoding all bitstreams in which either or both of the following conditions are true:

– profile_idc is equal to 77 or constraint_set1_flag is equal to 1 and the combination of level_idc and constraint_set3_flag represent a level less than or equal to the specified level.
– profile_idc is equal to 44, 100, 110, 122, or 244 and the value of level_idc represents a level less than or equal to the specified level.

A.2.8 High 10 Intra profile

Bitstreams conforming to the High 10 Intra profile shall obey the following constraints:

– All constraints specified in subclause A.2.5 for the High 10 profile shall be obeyed.
– All pictures shall be IDR pictures.
– Sequence parameter sets shall have num_ref_frames equal to 0.
– When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have num_reorder_frames equal to 0.
– When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have max_dec_frame_buffering equal to 0.
– Picture timing SEI messages, whether present in the bitstream (by non-VCL NAL units) or conveyed equivalently by other means not specified in this Recommendation | International Standard, shall have dpb_output_delay equal to 0.
– The level constraints specified for the High 10 Intra profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 10 Intra profile is specified by constraint_set3_flag being equal to 1 with profile_idc equal to 110. Decoders conforming to the High 10 Intra profile at a specific level shall be capable of decoding all bitstreams in which all of the following conditions are true.

– profile_idc is equal to 100 or 110.
– constraint_set3_flag is equal to 1.
– level_idc represents a level less than or equal to the specified level.

NOTE – The value 100 for profile_idc indicates that the bitstream conforms to the High profile as specified in subclause A.2.4. When profile_idc is equal to 100 and constraint_set3_flag is equal to 1, this indicates that the bitstream conforms to the High profile and additionally conforms to the constraints specified for the High 10 Intra profile in this subclause. For example, such a bitstream must have bit_depth_luma_minus8 equal to 0, have bit_depth_chroma_minus8 equal to 0, obey the MinCr, MaxBR and MaxCPB constraints of the High profile, contain only IDR pictures, have num_ref_frames equal to 0, have dpb_output_delay equal to 0, and obey the maximum slice size constraint of the High 10 Intra profile.

The operation of the deblocking filter process specified in subclause 8.7 is not required for decoder conformance to the High 10 Intra profile.

NOTE – The deblocking filter process specified in subclause 8.7 or some similar post-processing filter should be performed, although this is not a requirement for decoder conformance to the High 10 Intra profile. The syntax elements sent by an encoder for control of the deblocking filter process specified in subclause 8.7 are considered only as advisory information for decoders conformance to the High 10 Intra profile. However, the application of the deblocking filter process specified in subclause 8.7 is required for decoder conformance to the High 10, High 4:2:2, and High 4:4:4 Predictive profiles when decoding bitstreams that conform to the High 10 Intra profile.

A.2.9 High 4:2:2 Intra profile

Bitstreams conforming to the High 4:2:2 Intra profile shall obey the following constraints:

- All constraints specified in subclause A.2.6 for the High 4:2:2 profile shall be obeyed.
- All pictures shall be IDR pictures.
- Sequence parameter sets shall have num_ref_frames equal to 0.
- When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have num_reorder_frames equal to 0.
- When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have max_dec_frame_buffering equal to 0.
- Picture timing SEI messages, whether present in the bitstream (by non-VCL NAL units) or conveyed equivalently by other means not specified in this Recommendation | International Standard, shall have dpb_output_delay equal to 0.
- The level constraints specified for the High 4:2:2 Intra profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 4:2:2 Intra profile is specified by constraint_set3_flag being equal to 1 with profile_idc equal to 122. Decoders conforming to the High 4:2:2 Intra profile at a specific level shall be capable of decoding all bitstreams in which all of the following conditions are true:

- profile_idc is equal to 100, 110, or 122.
- constraint_set3_flag is equal to 1.
- level_idc represents a level less than or equal to the specified level.

The operation of the deblocking filter process specified in subclause 8.7 is not required for decoder conformance to the High 4:2:2 Intra profile.

NOTE – The deblocking filter process specified in subclause 8.7 or some similar post-processing filter should be performed, although this is not a requirement for decoder conformance to the High 4:2:2 Intra profile. The syntax elements sent by an encoder for control of the deblocking filter process specified in subclause 8.7 are considered only as advisory information for decoders conformance to the High 4:2:2 Intra profile. However, the application of the deblocking filter process specified in subclause 8.7 is required for decoder conformance to the High 4:2:2, and High 4:4:4 Predictive profiles when decoding bitstreams that conform to the High 4:2:2 Intra profile.

A.2.10 High 4:4:4 Intra profile

Bitstreams conforming to the High 4:4:4 Intra profile shall obey the following constraints:

- All constraints specified in subclause A.2.7 for the High 4:4:4 Predictive profile shall be obeyed.
- All pictures shall be IDR pictures.
- Sequence parameter sets shall have num_ref_frames equal to 0.
- When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have num_reorder_frames equal to 0.
- When vui_parameters_present_flag is equal to 1 and bitstream_restriction_flag is equal to 1, sequence parameter sets shall have max_dec_frame_buffering equal to 0.
- Picture timing SEI messages, whether present in the bitstream (by non-VCL NAL units) or conveyed equivalently by other means not specified in this Recommendation | International Standard, shall have dpb_output_delay equal to 0.
- The level constraints specified for the High 4:4:4 Intra profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the High 4:4:4 Intra profile is specified by constraint_set3_flag being equal to 1 with profile_idc equal to 244. Decoders conforming to the High 4:4:4 Intra profile at a specific level shall be capable of decoding all bitstreams in which all of the following conditions are true:

- profile_idc is equal to 44, 100, 110, 122, or 244.
- constraint_set3_flag is equal to 1.
- level_idc represents a level less than or equal to the specified level.

The operation of the deblocking filter process specified in subclause 8.7 is not required for decoder conformance to the High 4:4:4 Intra profile.

NOTE – The deblocking filter process specified in subclause 8.7 or some similar post-processing filter should be performed, although this is not a requirement for decoder conformance to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles. The syntax elements sent by an encoder for control of the deblocking filter process specified in subclause 8.7 are considered only as advisory information for decoders conformance to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles. However, the application of the

deblocking filter process specified in subclause 8.7 is required for decoder conformance to the High 4:4:4 Predictive profile when decoding bitstreams that conform to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles.

A.2.11 CAVLC 4:4:4 Intra profile

Bitstreams conforming to the CAVLC 4:4:4 Intra profile shall obey the following constraints:

– All constraints specified in subclause A.2.10 for the High 4:4:4 Intra profile shall be obeyed.

– Picture parameter sets shall have entropy_coding_mode_flag equal to 0.

– The level constraints specified for the CAVLC 4:4:4 Intra profile in subclause A.3 shall be fulfilled.

Conformance of a bitstream to the CAVLC 4:4:4 Intra profile is specified by profile_idc being equal to 44. Decoders conforming to the CAVLC 4:4:4 Intra profile at a specific level shall be capable of decoding all bitstreams in which all of the following conditions are true:

– profile_idc is equal to 44.

– level_idc represents a level less than or equal to the specified level.

The operation of the deblocking filter process specified in subclause 8.7 is not required for decoder conformance to the CAVLC 4:4:4 Intra profile.

NOTE – The deblocking filter process specified in subclause 8.7 or some similar post-processing filter should be performed, although this is not a requirement for decoder conformance to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles. The syntax elements sent by an encoder for control of the deblocking filter process specified in subclause 8.7 are considered only as advisory information for decoders conformance to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles. However, the application of the deblocking filter process specified in subclause 8.7 is required for decoder conformance to the High 4:4:4 Predictive profile when decoding bitstreams that conform to the High 4:4:4 Intra and CAVLC 4:4:4 Intra profiles.

A.3 Levels

The following is specified for expressing the constraints in this annex.

– Let access unit n be the n-th access unit in decoding order with the first access unit being access unit 0.

– Let picture n be the primary coded picture or the corresponding decoded picture of access unit n.

A.3.1 Level limits common to the Baseline, Main, and Extended profiles

Let the variable fR be derived as follows.

– If picture n is a frame, fR is set equal to 1 ÷ 172.

– Otherwise (picture n is a field), fR is set equal to 1 ÷ (172 * 2).

Bitstreams conforming to the Baseline, Main, or Extended profiles at a specified level shall obey the following constraints:

a) The nominal removal time of access unit n (with n > 0) from the CPB as specified in subclause C.1.2, satisfies the constraint that $t_{r,n}(n) - t_r(n - 1)$ is greater than or equal to Max(PicSizeInMbs ÷ MaxMBPS, fR), where MaxMBPS is the value specified in Table A-1 that applies to picture n – 1, and PicSizeInMbs is the number of macroblocks in picture n – 1.

b) The difference between consecutive output times of pictures from the DPB as specified in subclause C.2.2, satisfies the constraint that $\Delta t_{o,dpb}(n) >= Max(PicSizeInMbs \div MaxMBPS, fR)$, where MaxMBPS is the value specified in Table A-1 for picture n, and PicSizeInMbs is the number of macroblocks of picture n, provided that picture n is a picture that is output and is not the last picture of the bitstream that is output.

c) The sum of the NumBytesInNALunit variables for access unit 0 is less than or equal to $384 * (PicSizeInMbs + MaxMBPS * (t_r(0) - t_{r,n}(0))) \div MinCR$, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture 0 and PicSizeInMbs is the number of macroblocks in picture 0.

d) The sum of the NumBytesInNALunit variables for access unit n (with n > 0) is less than or equal to $384 * MaxMBPS * (t_r(n) - t_r(n - 1)) \div MinCR$, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture n.

e) PicWidthInMbs * FrameHeightInMbs <= MaxFS, where MaxFS is specified in Table A-1.

f) PicWidthInMbs <= Sqrt(MaxFS * 8).

g) FrameHeightInMbs <= Sqrt(MaxFS * 8).

h) max_dec_frame_buffering <= MaxDpbSize, where MaxDpbSize is equal to Min(1024 * MaxDPB / (PicWidthInMbs * FrameHeightInMbs * 384), 16) and MaxDPB is given in Table A-1 in units of 1024 bytes.

i) For the VCL HRD parameters, BitRate[SchedSelIdx] <= 1000 * MaxBR and CpbSize[SchedSelIdx] <= 1000 * MaxCPB for at least one value of SchedSelIdx, where BitRate[SchedSelIdx] is given by Equation E-37 and CpbSize[SchedSelIdx] is given by Equation E-38 when vcl_hrd_parameters_present_flag is equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of 1000 bits/s and 1000 bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1, inclusive.

j) For the NAL HRD parameters, BitRate[SchedSelIdx] <= 1200 * MaxBR and CpbSize[SchedSelIdx] <= 1200 * MaxCPB for at least one value of SchedSelIdx, where BitRate[SchedSelIdx] is given by Equation E-37 and CpbSize[SchedSelIdx] is given by Equation E-38 when nal_hrd_parameters_present_flag equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of 1200 bits/s and 1200 bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1.

k) Vertical motion vector component range luma motion vectors does not exceed MaxVmvR in units of luma frame samples, where MaxVmvR is specified in Table A-1.

NOTE 1 – When chroma_format_idc is equal to 1 and the current macroblock is a field macroblock, the motion vector component range for chroma motion vectors may exceed MaxVmvR in units of luma frame samples, due to the method of deriving chroma motion vectors as specified in subclause 8.4.1.4.

l) Horizontal motion vector range does not exceed the range of -2048 to 2047.75, inclusive, in units of luma samples.

m) Number of motion vectors per two consecutive macroblocks in decoding order (also applying to the total from the last macroblock of a slice and the first macroblock of the next slice in decoding order, and in particular also applying to the total from the last macroblock of the last slice of a picture and the first macroblock of the first slice of the next picture in decoding order) does not exceed MaxMvsPer2Mb, where MaxMvsPer2Mb is specified in Table A-1. The number of motion vectors for each macroblock is the value of the variable MvCnt after the completion of the intra or inter prediction process for the macroblock.

n) Number of bits of macroblock layer( ) data for any macroblock is not greater than 3200. Depending on entropy_coding_mode_flag, the bits of macroblock_layer( ) data are counted as follows.

- If entropy_coding_mode_flag is equal to 0, the number of bits of macroblock_layer( ) data is given by the number of bits in the macroblock_layer( ) syntax structure for a macroblock.
- Otherwise (entropy_coding_mode_flag is equal to 1), the number of bits of macroblock_layer( ) data for a macroblock is given by the number of times read_bits(1) is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing the macroblock_layer( ) associated with the macroblock.

Table A-1 specifies the limits for each level. Entries marked "-" in Table A-1 denote the absence of a corresponding limit. For purposes of comparison of level capabilities, a level shall be considered to be a lower (higher) level than some other level if the level appears nearer to the top (bottom) row of Table A-1 than the other level.

A level to which the bitstream conforms shall be indicated by the syntax elements level_idc and constraint_set3_flag as follows.

- If level_idc is equal to 11 and constraint_set3_flag is equal to 1, the indicated level is level 1b.
- Otherwise (level_idc is not equal to 11 or constraint_set3_flag is not equal to 1), level_idc shall be set equal to a value of ten times the level number specified in Table A-1 and constraint_set3_flag shall be set equal to 0.

Table A-1 – Level limits

Level number	Max macroblock processing rate MaxMBPS (MB/s)	Max frame size MaxFS (MBs)	Max decoded picture buffer size MaxDPB (1024 bytes for 4:2:0)	Max video bit rate MaxBR (1000 bits/s, 1200 bits/s, cpbBrVclFactor bits/s, or cpbBrNalFactor bits/s)	Max CPB size MaxCPB (1000 bits, 1200 bits, cpbBrVclFactor bits, or cpbBrNalFactor bits)	Vertical MV component range MaxVmvR (luma frame samples)	Min compression ratio MinCR	Max number of motion vectors per two consecutive MBs MaxMvsPer2Mb
1	1 485	99	148.5	64	175	[-64,+63.75]	2	-
1b	1 485	99	148.5	128	350	[-64,+63.75]	2	-
1.1	3 000	396	337.5	192	500	[-128,+127.75]	2	-
1.2	6 000	396	891.0	384	1 000	[-128,+127.75]	2	-
1.3	11 880	396	891.0	768	2 000	[-128,+127.75]	2	-
2	11 880	396	891.0	2 000	2 000	[-128,+127.75]	2	-
2.1	19 800	792	1 782.0	4 000	4 000	[-256,+255.75]	2	-
2.2	20 250	1 620	3 037.5	4 000	4 000	[-256,+255.75]	2	-
3	40 500	1 620	3 037.5	10 000	10 000	[-256,+255.75]	2	32
3.1	108 000	3 600	6 750.0	14 000	14 000	[-512,+511.75]	4	16
3.2	216 000	5 120	7 680.0	20 000	20 000	[-512,+511.75]	4	16
4	245 760	8 192	12 288.0	20 000	25 000	[-512,+511.75]	4	16
4.1	245 760	8 192	12 288.0	50 000	62 500	[-512,+511.75]	2	16
4.2	522 240	8 704	13 056.0	50 000	62 500	[-512,+511.75]	2	16
5	589 824	22 080	41 400.0	135 000	135 000	[-512,+511.75]	2	16
5.1	983 040	36 864	69 120.0	240 000	240 000	[-512,+511.75]	2	16

Levels with non-integer level numbers in Table A-1 are referred to as "intermediate levels".

NOTE 2 – All levels have the same status, but some applications may choose to use only the integer-numbered levels.

Informative subclause A.3.4 shows the effect of these limits on frame rates for several example picture formats.

A.3.2 Level limits common to the High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profiles

Let the variable fR be derived as follows.

– If picture n is a frame, fR is set equal to 1 ÷ 172.

– Otherwise (picture n is a field), fR is set equal to 1 ÷ (172 * 2).

Bitstreams conforming to the High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles at a specified level shall obey the following constraints:

a) The nominal removal time of access unit n (with n > 0) from the CPB as specified in subclause C.1.2, satisfies the constraint that $t_{r,n}(n) - t_r(n - 1)$ is greater than or equal to Max(PicSizeInMbs ÷ MaxMBPS, fR), where MaxMBPS is the value specified in Table A-1 that applies to picture n – 1, and PicSizeInMbs is the number of macroblocks in picture n – 1.

b) The difference between consecutive output times of pictures from the DPB as specified in subclause C.2.2, satisfies the constraint that $\Delta t_{o,dpb}(n) >= Max(PicSizeInMbs ÷ MaxMBPS, fR)$, where MaxMBPS is the value specified in Table A-1 for picture n, and PicSizeInMbs is the number of macroblocks of picture n, provided that picture n is a picture that is output and is not the last picture of the bitstream that is output.

c) PicWidthInMbs * FrameHeightInMbs <= MaxFS, where MaxFS is specified in Table A-1.

d) PicWidthInMbs <= Sqrt(MaxFS * 8).

e) FrameHeightInMbs <= Sqrt(MaxFS * 8).

f) max_dec_frame_buffering <= MaxDpbSize, where MaxDpbSize is equal to Min(1024 * MaxDPB / (PicWidthInMbs * FrameHeightInMbs * 384), 16) and MaxDPB is specified in Table A-1.

g) Vertical motion vector component range does not exceed MaxVmvR in units of luma frame samples, where MaxVmvR is specified in Table A-1.

h) Horizontal motion vector range does not exceed the range of -2048 to 2047.75, inclusive, in units of luma samples.

i) Number of motion vectors per two consecutive macroblocks in decoding order (also applying to the total from the last macroblock of a slice and the first macroblock of the next slice in decoding order) does not exceed MaxMvsPer2Mb, where MaxMvsPer2Mb is specified in Table A-1. The number of motion vectors for each macroblock is value of the variable MvCnt after the completion of the intra or inter prediction process for the macroblock.

j) Number of bits of macroblock_layer( ) data for any macroblock is not greater than 128 + RawMbBits. Depending on entropy_coding_mode_flag, the bits of macroblock_layer( ) data are counted as follows.

   – If entropy_coding_mode_flag is equal to 0, the number of bits of macroblock_layer( ) data is given by the number of bits in the macroblock_layer( ) syntax structure for a macroblock.

   – Otherwise (entropy_coding_mode_flag is equal to 1), the number of bits of macroblock_layer( ) data for a macroblock is given by the number of times read_bits(1) is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing the macroblock_layer( ) associated with the macroblock.

Table A-1 specifies the limits for each level. Entries marked "-" in Table A-1 denote the absence of a corresponding limit. The use of the MinCR parameter column of Table A-1 for these profiles is specified in subclause A.3.3.

A level to which the bitstream conforms shall be indicated by the syntax element level_idc as follows.

– If level_idc is equal to 9, the indicated level is level 1b.

– Otherwise (level_idc is not equal to 9), level_idc shall be set equal to a value of ten times the level number specified in Table A-1.

A.3.3 Profile-specific level limits

a) In bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, the removal time of access unit 0 shall satisfy the constraint that the number of slices in picture 0 is less than or equal to (PicSizeInMbs + MaxMBPS * (t_r(0) - $t_{r,n}$(0))) ÷ SliceRate, where SliceRate is the value specified in Table A-4 that applies to picture 0.

b) In bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, the difference between consecutive removal time of access units n and n - 1 (with n > 0) shall satisfy the constraint that the number of slices in picture n is less than or equal to MaxMBPS * (t_r(n) - t_r(n - 1)) ÷ SliceRate, where SliceRate is the value specified in Table A-4 that applies to picture n.

c) In bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, sequence parameter sets shall have direct_8x8_inference_flag equal to 1 for the levels specified in Table A-4.

   NOTE 1 – direct_8x8_inference_flag is not relevant to the Baseline profile as it does not allow B slice types (specified in subclause A.2.1), and direct_8x8_inference_flag is equal to 1 for all levels of the Extended profile (specified in subclause A.2.3).

d) In bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, CAVLC 4:4:4 Intra, or Extended profiles, sequence parameter sets shall have frame_mbs_only_flag equal to 1 for the levels specified in Table A-4 for the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles and in Table A-5 for the Extended profile.

   NOTE 2 – frame_mbs_only_flag is equal to 1 for all levels of the Baseline profile (specified in subclause A.2.1).

e) In bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, CAVLC 4:4:4 Intra, or Extended profiles, the value of sub_mb_type in B macroblocks shall not be equal to B_Bi_8x4, B_Bi_4x8, or B_Bi_4x4 for the levels in which MinLumaBiPredSize is shown as 8x8 in Table A-4 for the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles and in Table A-5 for the Extended profile.

f) In bitstreams conforming to the Baseline or Extended profiles, ($xInt_{max} - xInt_{min} + 6$) * ($yInt_{max} - yInt_{min} + 6$) <= MaxSubMbRectSize in macroblocks coded with mb_type equal to P_8x8, P_8x8ref0 or B_8x8 for all invocations of the process specified in subclause 8.4.2.2.1 used to generate the predicted luma sample array for a single reference picture list (reference picture list 0 or reference picture list 1) for each 8x8 sub-macroblock, where NumSubMbPart(sub_mb_type) > 1, where MaxSubMbRectSize is specified in Table A-3 for the Baseline profile and in Table A-5 for the Extended profile and

- $xInt_{min}$ is the minimum value of $xInt_L$ among all luma sample predictions for the sub-macroblock,
- $xInt_{max}$ is the maximum value of $xInt_L$ among all luma sample predictions for the sub-macroblock,
- $yInt_{min}$ is the minimum value of $yInt_L$ among all luma sample predictions for the sub-macroblock,
- $yInt_{max}$ is the maximum value of $yInt_L$ among all luma sample predictions for the sub-macroblock.

g) In bitstreams conforming to the High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, for the VCL HRD parameters, BitRate[SchedSelIdx] <= cpbBrVclFactor * MaxBR and CpbSize[SchedSelIdx] <= cpbBrVclFactor * MaxCPB for at least one value of SchedSelIdx, where cpbBrVclFactor is specified in Table A-2, BitRate[SchedSelIdx] is specified by Equation E-37 and CpbSize[SchedSelIdx] is specified by Equation E-38 when vcl_hrd_parameters_present_flag is equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of cpbBrVclFactor bits/s and cpbBrVclFactor bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1, inclusive.

h) In bitstreams conforming to the High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, for the NAL HRD parameters, BitRate[SchedSelIdx] <= cpbBrNalFactor * MaxBR and CpbSize[SchedSelIdx] <= cpbBrNalFactor * MaxCPB for at least one value of SchedSelIdx, where cpbBrNalFactor is specified in Table A-2, BitRate[SchedSelIdx] is specified by Equation E-37 and CpbSize[SchedSelIdx] is specified by Equation E-38 when nal_hrd_parameters_present_flag equal to 1. MaxBR and MaxCPB are specified in Table A-1 in units of cpbBrNalFactor bits/s and cpbBrNalFactor bits, respectively. The bitstream shall satisfy these conditions for at least one value of SchedSelIdx in the range 0 to cpb_cnt_minus1.

i) In bitstreams conforming to the High profile, the sum of the NumBytesInNALunit variables for access unit 0 is less than or equal to 384 * (PicSizeInMbs + MaxMBPS * ($t_r(0) - t_{r,n}(0)$)) ÷ MinCR, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture 0 and PicSizeInMbs is the number of macroblocks in picture 0.

> NOTE 2 – Such a limit involving MinCR is not imposed for bitstream conformance to the High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profiles.

j) In bitstreams conforming to the High profile, the sum of the NumBytesInNALunit variables for access unit n (with n > 0) is less than or equal to 384 * MaxMBPS * ($t_r(n) - t_r(n-1)$) ÷ MinCR, where MaxMBPS and MinCR are the values specified in Table A-1 that apply to picture n.

> NOTE 3 – Such a limit involving MinCR is not imposed for bitstream conformance to the High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profiles.

k) In bitstreams conforming to the High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles, when PicSizeInMbs is greater than 1620, the number of macroblocks in any coded slice shall not exceed MaxFS / 4, where MaxFS is specified in Table A-1.

> NOTE 4 – Items i), j), and k) above are included herein for purposes of specification of the High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profiles. The aspects of these items as they relate to the High, High 10, and High 4:2:2 profiles are included herein for completeness, and reflect corrections of the intended specification of these profiles.

Table A-2 – Specification of cpbBrVclFactor and cpbBrNalFactor

Profile	cpbBrVclFactor	cpbBrNalFactor
High	1 250	1 500
High 10 High 10 Intra	3 000	3 600
High 4:2:2 High 4:2:2 Intra	4 000	4 800
High 4:4:4 Predictive High 4:4:4 Intra CAVLC 4:4:4 Intra	4 000	4 800

A.3.3.1 Baseline profile level limits

Table A-3 specifies limits for each level that are specific to bitstreams conforming to the Baseline profile. Entries marked "-" in Table A-3 denote the absence of a corresponding limit.

Table A-3 – Baseline profile level limits

Level number	MaxSubMbRectSize
1	576
1b	576
1.1	576
1.2	576
1.3	576
2	576
2.1	576
2.2	576
3	576
3.1	-
3.2	-
4	-
4.1	-
4.2	-
5	-
5.1	-

A.3.3.2 Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profile level limits

Table A-4 specifies limits for each level that are specific to bitstreams conforming to the Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, or CAVLC 4:4:4 Intra profiles. Entries marked "-" in Table A-4 denote the absence of a corresponding limit.

Table A-4 – Main, High, High 10, High 4:2:2, High 4:4:4 Predictive, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, and CAVLC 4:4:4 Intra profile level limits

Level number	SliceRate	MinLumaBiPredSize	direct_8x8_inference_flag	frame_mbs_only_flag
1	-	-	-	1
1b	-	-	-	1
1.1	-	-	-	1
1.2	-	-	-	1
1.3	-	-	-	1
2	-	-	-	1
2.1	-	-	-	-
2.2	-	-	-	-
3	22	-	1	-
3.1	60	8x8	1	-
3.2	60	8x8	1	-
4	60	8x8	1	-
4.1	24	8x8	1	-
4.2	24	8x8	1	1
5	24	8x8	1	1
5.1	24	8x8	1	1

A.3.3.3 Extended Profile level limits

Table A-5 specifies limits for each level that are specific to bitstreams conforming to the Extended profile. Entries marked "-" in Table A-5 denote the absence of a corresponding limit.

Table A-5 – Extended profile level limits

Level number	MaxSubMbRectSize	MinLumaBiPredSize	frame_mbs_only_flag
1	576	-	1
1b	576	-	1
1.1	576	-	1
1.2	576	-	1
1.3	576	-	1
2	576	-	1
2.1	576	-	-
2.2	576	-	-
3	576	-	-
3.1	-	8x8	-
3.2	-	8x8	-
4	-	8x8	-
4.1	-	8x8	-
4.2	-	8x8	1
5	-	8x8	1
5.1	-	8x8	1

A.3.4 Effect of level limits on frame rate (informative)

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

Table A-6 – Maximum frame rates (frames per second) for some example frame sizes

Level:					1	1b	1.1	1.2	1.3	2	2.1
Max frame size (macroblocks):					99	99	396	396	396	396	792
Max macroblocks/second:					1 485	1 485	3 000	6 000	11 880	11 880	19 800
Max frame size (samples):					25 344	25 344	101 376	101 376	101 376	101 376	202 752
Max samples/second:					380 160	380 160	768 000	1 536 000	3 041 280	3 041 280	5 068 800
Format	Luma Width	Luma Height	MBs Total	Luma Samples							
SQCIF	128	96	48	12 288	30.9	30.9	62.5	125.0	172.0	172.0	172.0
QCIF	176	144	99	25 344	15.0	15.0	30.3	60.6	120.0	120.0	172.0
QVGA	320	240	300	76 800	-	-	10.0	20.0	39.6	39.6	66.0
525 SIF	352	240	330	84 480	-	-	9.1	18.2	36.0	36.0	60.0
CIF	352	288	396	101 376	-	-	7.6	15.2	30.0	30.0	50.0
525 HHR	352	480	660	168 960	-	-	-	-	-	-	30.0
625 HHR	352	576	792	202 752	-	-	-	-	-	-	25.0
VGA	640	480	1 200	307 200	-	-	-	-	-	-	-
525 4SIF	704	480	1 320	337 920	-	-	-	-	-	-	-
525 SD	720	480	1 350	345 600	-	-	-	-	-	-	-
4CIF	704	576	1 584	405 504	-	-	-	-	-	-	-
625 SD	720	576	1 620	414 720	-	-	-	-	-	-	-
SVGA	800	600	1 900	486 400	-	-	-	-	-	-	-
XGA	1024	768	3 072	786 432	-	-	-	-	-	-	-
720p HD	1280	720	3 600	921 600	-	-	-	-	-	-	-
4VGA	1280	960	4 800	1 228 800	-	-	-	-	-	-	-
SXGA	1280	1024	5 120	1 310 720	-	-	-	-	-	-	-
525 16SIF	1408	960	5 280	1 351 680	-	-	-	-	-	-	-
16CIF	1408	1152	6 336	1 622 016	-	-	-	-	-	-	-
4SVGA	1600	1200	7 500	1 920 000	-	-	-	-	-	-	-
1080 HD	1920	1088	8 160	2 088 960	-	-	-	-	-	-	-
2Kx1K	2048	1024	8 192	2 097 152	-	-	-	-	-	-	-
2Kx1080	2048	1088	8 704	2 228 224	-	-	-	-	-	-	-
4XGA	2048	1536	12 288	3 145 728	-	-	-	-	-	-	-
16VGA	2560	1920	19 200	4 915 200	-	-	-	-	-	-	-
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	-	-	-	-	-	-	-
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	-	-	-	-	-	-	-
4Kx2K	4096	2048	32 768	8 388 608	-	-	-	-	-	-	-
4096x2304 (16:9)	4096	2304	36 864	9 437 184	-	-	-	-	-	-	-

Table A-6 (continued) – Maximum frame rates (frames per second) for some example frame sizes

Level:					2.2	3	3.1	3.2	4	4.1	4.2
Max frame size (macroblocks):					1 620	1 620	3 600	5 120	8 192	8 192	8 704
Max macroblocks/second:					20 250	40 500	108 000	216 000	245 760	245 760	522 240
Max frame size (samples):					414 720	414 720	921 600	1 310 720	2 097 152	2 097 152	2 228 224
Max samples/second:					5 184 000	10 368 000	27 648 000	55 296 000	62 914 560	62 914 560	133 693 440
Format	Luma Width	Luma Height	MBs Total	Luma Samples							
SQCIF	128	96	48	12 288	172.0	172.0	172.0	172.0	172.0	172.0	172.0
QCIF	176	144	99	25 344	172.0	172.0	172.0	172.0	172.0	172.0	172.0
QVGA	320	240	300	76 800	67.5	135.0	172.0	172.0	172.0	172.0	172.0
525 SIF	352	240	330	84 480	61.4	122.7	172.0	172.0	172.0	172.0	172.0
CIF	352	288	396	101 376	51.1	102.3	172.0	172.0	172.0	172.0	172.0
525 HHR	352	480	660	168 960	30.7	61.4	163.6	172.0	172.0	172.0	172.0
625 HHR	352	576	792	202 752	25.6	51.1	136.4	172.0	172.0	172.0	172.0
VGA	640	480	1 200	307 200	16.9	33.8	90.0	172.0	172.0	172.0	172.0
525 4SIF	704	480	1 320	337 920	15.3	30.7	81.8	163.6	172.0	172.0	172.0
525 SD	720	480	1 350	345 600	15.0	30.0	80.0	160.0	172.0	172.0	172.0
4CIF	704	576	1 584	405 504	12.8	25.6	68.2	136.4	155.2	155.2	172.0
625 SD	720	576	1 620	414 720	12.5	25.0	66.7	133.3	151.7	151.7	172.0
SVGA	800	600	1 900	486 400	-	-	56.8	113.7	129.3	129.3	172.0
XGA	1024	768	3 072	786 432	-	-	35.2	70.3	80.0	80.0	172.0
720p HD	1280	720	3 600	921 600	-	-	30.0	60.0	68.3	68.3	145.1
4VGA	1280	960	4 800	1 228 800	-	-	-	45.0	51.2	51.2	108.8
SXGA	1280	1024	5 120	1 310 720	-	-	-	42.2	48.0	48.0	102.0
525 16SIF	1408	960	5 280	1 351 680	-	-	-	-	46.5	46.5	98.9
16CIF	1408	1152	6 336	1 622 016	-	-	-	-	38.8	38.8	82.4
4SVGA	1600	1200	7 500	1 920 000	-	-	-	-	32.8	32.8	69.6
1080 HD	1920	1088	8 160	2 088 960	-	-	-	-	30.1	30.1	64.0
2Kx1K	2048	1024	8 192	2 097 152	-	-	-	-	30.0	30.0	63.8
2Kx1080	2048	1088	8 704	2 228 224	-	-	-	-	-	-	60.0
4XGA	2048	1536	12 288	3 145 728	-	-	-	-	-	-	-
16VGA	2560	1920	19 200	4 915 200	-	-	-	-	-	-	-
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	-	-	-	-	-	-	-
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	-	-	-	-	-	-	-
4Kx2K	4096	2048	32 768	8 388 608	-	-	-	-	-	-	-
4096x2304 (16:9)	4096	2304	36 864	9 437 184	-	-	-	-	-	-	-

Table A-6 (concluded) – Maximum frame rates (frames per second) for some example frame sizes

Level:					5	5.1
Max frame size (macroblocks):					22 080	36 864
Max macroblocks/second:					589 824	983 040
Max frame size (samples):					5 652 480	9 437 184
Max samples/second:					150 994 944	251 658 240
Format	Luma Width	Luma Height	MBs Total	Luma Samples		
SQCIF	128	96	48	12 288	172.0	172.0
QCIF	176	144	99	25 344	172.0	172.0
QVGA	320	240	300	76 800	172.0	172.0
525 SIF	352	240	330	84 480	172.0	172.0
CIF	352	288	396	101 376	172.0	172.0
525 HHR	352	480	660	168 960	172.0	172.0
625 HHR	352	576	792	202 752	172.0	172.0
VGA	640	480	1 200	307 200	172.0	172.0
525 4SIF	704	480	1 320	337 920	172.0	172.0
525 SD	720	480	1 350	345 600	172.0	172.0
4CIF	704	576	1 584	405 504	172.0	172.0
625 SD	720	576	1 620	414 720	172.0	172.0
SVGA	800	600	1 900	486 400	172.0	172.0
XGA	1024	768	3 072	786 432	172.0	172.0
720p HD	1280	720	3 600	921 600	163.8	172.0
4VGA	1280	960	4 800	1 228 800	122.9	172.0
SXGA	1280	1024	5 120	1 310 720	115.2	172.0
525 16SIF	1408	960	5 280	1 351 680	111.7	172.0
16CIF	1408	1152	6 336	1 622 016	93.1	155.2
4SVGA	1600	1200	7 500	1 920 000	78.6	131.1
1080 HD	1920	1088	8 160	2 088 960	72.3	120.5
2Kx1K	2048	1024	8 192	2 097 152	72.0	120.0
2Kx1080	2048	1088	8 704	2 228 224	67.8	112.9
4XGA	2048	1536	12 288	3 145 728	48.0	80.0
16VGA	2560	1920	19 200	4 915 200	30.7	51.2
3616x1536 (2.35:1)	3616	1536	21 696	5 554 176	27.2	45.3
3672x1536 (2.39:1)	3680	1536	22 080	5 652 480	26.7	44.5
4Kx2K	4096	2048	32 768	8 388 608	-	30.0
4096x2304 (16:9)	4096	2304	36 864	9 437 184	-	26.7

The following should be noted.

- This Recommendation | International Standard is a variable-frame-size specification. The specific frame sizes in Table A-6 are illustrative examples only.
- As used in Table A-6, "525" refers to typical use for environments using 525 analogue scan lines (of which approximately 480 lines contain the visible picture region), and "625" refers to environments using 625 analogue scan lines (of which approximately 576 lines contain the visible picture region).
- XGA is also known as (aka) XVGA, 4SVGA aka UXGA, 16XGA aka 4Kx3K, CIF aka 625 SIF, 625 HHR aka 2CIF aka half 625 D-1, aka half 625 ITU-R BT.601, 525 SD aka 525 D-1 aka 525 ITU-R BT.601, 625 SD aka 625 D-1 aka 625 ITU-R BT.601.
- Frame rates given are correct for progressive scan modes. The frame rates are also correct for interlaced video coding for the cases of frame height divisible by 32.

Annex B
Byte stream format

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

This annex specifies syntax and semantics of a byte stream format specified for use by applications that deliver some or all of the NAL unit stream as an ordered stream of bytes or bits within which the locations of NAL unit boundaries need to be identifiable from patterns in the data, such as ITU-T Rec. H.222.0 | ISO/IEC 13818-1 systems or ITU-T Rec. H.320 systems. For bit-oriented delivery, the bit order for the byte stream format is specified to start with the MSB of the first byte, proceed to the LSB of the first byte, followed by the MSB of the second byte, etc.

The byte stream format consists of a sequence of byte stream NAL unit syntax structures. Each byte stream NAL unit syntax structure contains one start code prefix followed by one nal_unit(NumBytesInNALunit) syntax structure. It may (and under some circumstances, it shall) also contain an additional zero_byte syntax element. It may also contain one or more additional trailing_zero_8bits syntax elements. When it is the first byte stream NAL unit in the bitstream, it may also contain one or more additional leading_zero_8bits syntax elements.

B.1 Byte stream NAL unit syntax and semantics

B.1.1 Byte stream NAL unit syntax

byte_stream_nal_unit(NumBytesInNALunit) {	C	Descriptor
while(next_bits(24) != 0x000001 && next_bits(32) != 0x00000001)		
leading_zero_8bits /* equal to 0x00 */		f(8)
if(next_bits(24) != 0x000001)		
zero_byte /* equal to 0x00 */		f(8)
start_code_prefix_one_3bytes /* equal to 0x000001 */		f(24)
nal_unit(NumBytesInNALunit)		
while(more_data_in_byte_stream( ) && next_bits(24) != 0x000001 && next_bits(32) != 0x00000001)		
trailing_zero_8bits /* equal to 0x00 */		f(8)
}		

B.1.2 Byte stream NAL unit semantics

The order of byte stream NAL units in the byte stream shall follow the decoding order of the NAL units contained in the byte stream NAL units (see subclause 7.4.1.2). The content of each byte stream NAL unit is associated with the same access unit as the NAL unit contained in the byte stream NAL unit (see subclause 7.4.1.2.3).

leading_zero_8bits is a byte equal to 0x00.

NOTE – The leading_zero_8bits syntax element can only be present in the first byte stream NAL unit of the bitstream, because (as shown in the syntax diagram of subclause B.1.1) any bytes equal to 0x00 that follow a NAL unit syntax structure and precede the four-byte sequence 0x00000001 (which is to be interpreted as a zero_byte followed by a start_code_prefix_one_3bytes) will be considered to be trailing_zero_8bits syntax elements that are part of the preceding byte stream NAL unit.

zero_byte is a single byte equal to 0x00.

When any of the following conditions are fulfilled, the zero_byte syntax element shall be present.

– the nal_unit_type within the nal_unit( ) is equal to 7 (sequence parameter set) or 8 (picture parameter set),

– the byte stream NAL unit syntax structure contains the first NAL unit of an access unit in decoding order, as specified by subclause 7.4.1.2.3.

start_code_prefix_one_3bytes is a fixed-value sequence of 3 bytes equal to 0x000001. This syntax element is called a start code prefix.

trailing_zero_8bits is a byte equal to 0x00.

B.2 Byte stream NAL unit decoding process

Input to this process consists of an ordered stream of bytes consisting of a sequence of byte stream NAL unit syntax structures.

Output of this process consists of a sequence of NAL unit syntax structures.

At the beginning of the decoding process, the decoder initialises its current position in the byte stream to the beginning of the byte stream. It then extracts and discards each leading_zero_8bits syntax element (if present), moving the current position in the byte stream forward one byte at a time, until the current position in the byte stream is such that the next four bytes in the bitstream form the four-byte sequence 0x00000001.

The decoder then performs the following step-wise process repeatedly to extract and decode each NAL unit syntax structure in the byte stream until the end of the byte stream has been encountered (as determined by unspecified means) and the last NAL unit in the byte stream has been decoded:

1. When the next four bytes in the bitstream form the four-byte sequence 0x00000001, the next byte in the byte stream (which is a zero_byte syntax element) is extracted and discarded and the current position in the byte stream is set equal to the position of the byte following this discarded byte.
2. The next three-byte sequence in the byte stream (which is a start_code_prefix_one_3bytes) is extracted and discarded and the current position in the byte stream is set equal to the position of the byte following this three-byte sequence.
3. NumBytesInNALunit is set equal to the number of bytes starting with the byte at the current position in the byte stream up to and including the last byte that precedes the location of any of the following conditions:
    a. A subsequent byte-aligned three-byte sequence equal to 0x000000, or
    b. A subsequent byte-aligned three-byte sequence equal to 0x000001, or
    c. The end of the byte stream, as determined by unspecified means.
4. NumBytesInNALunit bytes are removed from the bitstream and the current position in the byte stream is advanced by NumBytesInNALunit bytes. This sequence of bytes is nal_unit(NumBytesInNALunit) and is decoded using the NAL unit decoding process.
5. When the current position in the byte stream is not at the end of the byte stream (as determined by unspecified means) and the next bytes in the byte stream do not start with a three-byte sequence equal to 0x000001 and the next bytes in the byte stream do not start with a four byte sequence equal to 0x00000001, the decoder extracts and discards each trailing_zero_8bits syntax element, moving the current position in the byte stream forward one byte at a time, until the current position in the byte stream is such that the next bytes in the byte stream form the four-byte sequence 0x00000001 or the end of the byte stream has been encountered (as determined by unspecified means).

B.3 Decoder byte-alignment recovery (informative)

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

Many applications provide data to a decoder in a manner that is inherently byte aligned, and thus have no need for the bit-oriented byte alignment detection procedure described in this subclause.

A decoder is said to have byte-alignment with a bitstream when the decoder is able to determine whether or not the positions of data in the bitstream are byte-aligned. When a decoder does not have byte alignment with the encoder's byte stream, the decoder may examine the incoming bitstream for the binary pattern '00000000 00000000 00000000 00000001' (31 consecutive bits equal to 0 followed by a bit equal to 1). The bit immediately following this pattern is the first bit of an aligned byte following a start code prefix. Upon detecting this pattern, the decoder will be byte aligned with the encoder and positioned at the start of a NAL unit in the byte stream.

Once byte aligned with the encoder, the decoder can examine the incoming byte stream for subsequent three-byte sequences 0x000001 and 0x000003.

When the three-byte sequence 0x000001 is detected, this is a start code prefix.

When the three-byte sequence 0x000003 is detected, the third byte (0x03) is an emulation_prevention_three_byte to be discarded as specified in subclause 7.4.1.

When an error in the bitstream syntax is detected (e.g., a non-zero value of the forbidden_zero_bit or one of the three-byte or four-byte sequences that are prohibited in subclause 7.4.1), the decoder may consider the detected condition as an indication that byte alignment may have been lost and may discard all bitstream data until the detection of byte alignment at a later position in the bitstream as described in this subclause.

Annex C
Hypothetical reference decoder

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

This annex specifies the hypothetical reference decoder (HRD) and its use to check bitstream and decoder conformance.

Two types of bitstreams are subject to HRD conformance checking for this Recommendation | International Standard. The first such type of bitstream, called Type I bitstream, is a NAL unit stream containing only the VCL NAL units and filler data NAL units for all access units in the bitstream. The second type of bitstream, called a Type II bitstream, contains, in addition to the VCL NAL units and filler data NAL units for all access units in the bitstream, at least one of the following.

- additional non-VCL NAL units other than filler data NAL units.
- all leading_zero_8bits, zero_byte, start_code_prefix_one_3bytes, and trailing_zero_8bits syntax elements that form a byte stream from the NAL unit stream (as specified in Annex B).

Figure C-1 shows the types of bitstream conformance points checked by the HRD.

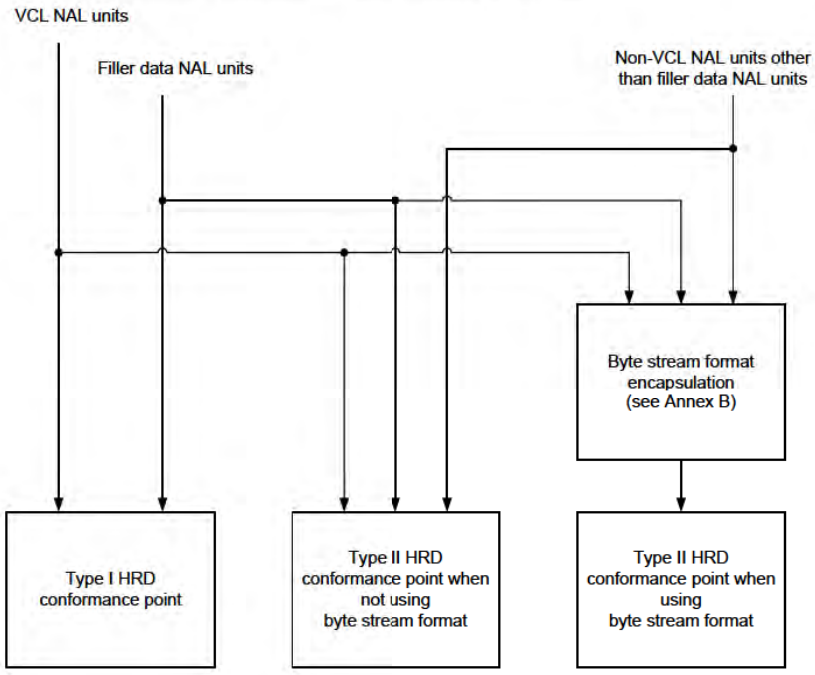


Figure C-1 – Structure of byte streams and NAL unit streams for HRD conformance checks

The syntax elements of non-VCL NAL units (or their default values for some of the syntax elements), required for the HRD, are specified in the semantic subclauses of clause 7 and Annexes D and E.

Two types of HRD parameter sets are used. The HRD parameter sets are signalled through video usability information as specified in subclauses E.1 and E.2, which is part of the sequence parameters set syntax structure.

In order to check conformance of a bitstream using the HRD, all sequence parameter sets and picture parameters sets referred to in the VCL NAL units, and corresponding buffering period and picture timing SEI messages shall be conveyed to the HRD, in a timely manner, either in the bitstream (by non-VCL NAL units), or by other means not specified in this Recommendation | International Standard.

In Annexes C, D and E, the specification for "presence" of non-VCL NAL units is also satisfied when those NAL units (or just some of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

NOTE 1 – As an example, synchronization of a non-VCL NAL unit, conveyed by means other than presence in the bitstream, with the NAL units that are present in the bitstream, can be achieved by indicating two points in the bitstream, between which the non-VCL NAL unit would have been present in the bitstream, had the encoder decided to convey it in the bitstream.

When the content of a non-VCL NAL unit is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the non-VCL NAL unit is not required to use the same syntax specified in this annex.

NOTE 2 – When HRD information is contained within the bitstream, it is possible to verify the conformance of a bitstream to the requirements of this subclause based solely on information contained in the bitstream. When the HRD information is not present in the bitstream, as is the case for all "stand-alone" Type I bitstreams, conformance can only be verified when the HRD data is supplied by some other means not specified in this Recommendation | International Standard.

The HRD contains a coded picture buffer (CPB), an instantaneous decoding process, a decoded picture buffer (DPB), and output cropping as shown in Figure C-2.

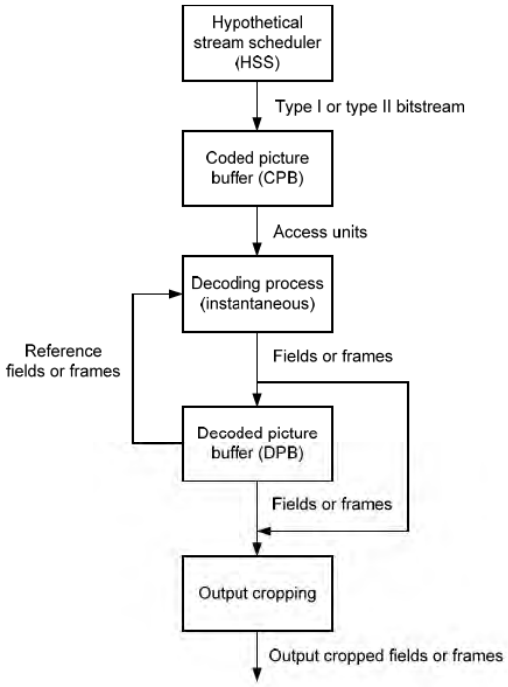


Figure C-2 – HRD buffer model

The CPB size (number of bits) is CpbSize[SchedSelIdx]. The DPB size (number of frame buffers) is Max(1, max_dec_frame_buffering).

The HRD operates as follows. Data associated with access units that flow into the CPB according to a specified arrival schedule are delivered by the HSS. The data associated with each access unit are removed and decoded instantaneously by the instantaneous decoding process at CPB removal times. Each decoded picture is placed in the DPB at its CPB removal time unless it is output at its CPB removal time and is a non-reference picture. When a picture is placed in the DPB it is removed from the DPB at the later of the DPB output time or the time that it is marked as "unused for reference".

For each picture in the bitstream, the variable OutputFlag for the decoded picture and, when applicable, the reference base picture is set as follows.

– If the coded video sequence containing the picture conforms to one or more of the profiles specified in Annex A and the decoding process specified in clauses 2-9 is used), OutputFlag is set equal to 1.

– Otherwise (the coded video sequence containing the picture conforms to a profile specified in Annex G and the decoding process specified in Annex G is used), the following applies.
  – For a reference base picture, OutputFlag is set equal to 0.
  – For a decoded picture, OutputFlag is set equal to the value of the output_flag syntax element of the target layer representation.

The operation of the CPB is specified in subclause C.1. The instantaneous decoder operation is specified in clauses 8 and 9. The operation of the DPB is specified in subclause C.2. The output cropping is specified in subclause C.2.2.

HSS and HRD information concerning the number of enumerated delivery schedules and their associated bit rates and buffer sizes is specified in subclauses E.1.1, E.1.2, E.2.1 and E.2.2. The HRD is initialised as specified by the buffering period SEI message as specified in subclauses D.1.1 and D.2.1. The removal timing of access units from the CPB and output timing from the DPB are specified in the picture timing SEI message as specified in subclauses D.1.2 and D.2.2. All timing information relating to a specific access unit shall arrive prior to the CPB removal time of the access unit.

The HRD is used to check conformance of bitstreams and decoders as specified in subclauses C.3 and C.4, respectively.

NOTE 3 – While conformance is guaranteed under the assumption that all frame-rates and clocks used to generate the bitstream match exactly the values signalled in the bitstream, in a real system each of these may vary from the signalled or specified value.

All the arithmetic in this annex is done with real values, so that no rounding errors can propagate. For example, the number of bits in a CPB just prior to or after removal of an access unit is not necessarily an integer.

The variable t_c is derived as follows and is called a clock tick.

$$t_c = num\_units\_in\_tick \div time\_scale \quad (C\text{-}1)$$

The following is specified for expressing the constraints in this annex.

– Let access unit n be the n-th access unit in decoding order with the first access unit being access unit 0.

– Let picture n be the primary coded picture or the decoded primary picture of access unit n.

C.1 Operation of coded picture buffer (CPB)

The specifications in this subclause apply independently to each set of CPB parameters that is present and to both the Type I and Type II conformance points shown in Figure C-1.

C.1.1 Timing of bitstream arrival

The HRD may be initialised at any one of the buffering period SEI messages. Prior to initialisation, the CPB is empty.

NOTE – After initialisation, the HRD is not initialised again by subsequent buffering period SEI messages.

Each access unit is referred to as access unit n, where the number n identifies the particular access unit. The access unit that is associated with the buffering period SEI message that initialises the CPB is referred to as access unit 0. The value of n is incremented by 1 for each subsequent access unit in decoding order.

The time at which the first bit of access unit n begins to enter the CPB is referred to as the initial arrival time $t_{ai}(n)$.

The initial arrival time of access units is derived as follows.

– If the access unit is access unit 0, $t_{ai}(0) = 0$,

  Otherwise (the access unit is access unit n with n > 0), the following applies.

  – If cbr_flag[SchedSelIdx] is equal to 1, the initial arrival time for access unit n, is equal to the final arrival time (which is derived below) of access unit n - 1, i.e.,

$$t_{ai}(n) = t_{af}(n - 1) \quad (C\text{-}2)$$

  – Otherwise (cbr_flag[SchedSelIdx] is equal to 0), the initial arrival time for access unit n is derived by

$$t_{ai}(n) = Max(t_{af}(n - 1), t_{ai,earliest}(n)) \quad (C\text{-}3)$$

  where $t_{ai,earliest}(n)$ is derived as follows

– If access unit n is not the first access unit of a subsequent buffering period, $t_{ai,earliest}(n)$ is derived as

$$t_{ai,earliest}(n) = t_{r,n}(n) - (initial\_cpb\_removal\_delay[SchedSelIdx] + initial\_cpb\_removal\_delay\_offset[SchedSelIdx]) \div 90000 \quad (C\text{-}4)$$

with $t_{r,n}(n)$ being the nominal removal time of access unit n from the CPB as specified in subclause C.1.2 and initial_cpb_removal_delay[SchedSelIdx] and initial_cpb_removal_delay_offset[SchedSelIdx] being specified in the previous buffering period SEI message.

– Otherwise (access unit n is the first access unit of a subsequent buffering period), $t_{ai,earliest}(n)$ is derived as

$$t_{ai,earliest}(n) = t_{r,n}(n) - (initial\_cpb\_removal\_delay[SchedSelIdx] \div 90000) \quad (C\text{-}5)$$

with initial_cpb_removal_delay[SchedSelIdx] being specified in the buffering period SEI message associated with access unit n.

The final arrival time for access unit n is derived by

$$t_{af}(n) = t_{ai}(n) + b(n) \div BitRate[SchedSelIdx] \quad (C\text{-}6)$$

where b(n) is the size in bits of access unit n, counting the bits of the VCL NAL units and the filler data NAL units for the Type I conformance point or all bits of the Type II bitstream for the Type II conformance point, where the Type I and Type II conformance points are as shown in Figure C-1.

The values of SchedSelIdx, BitRate[SchedSelIdx], and CpbSize[SchedSelIdx] are constrained as follows.

– If the content of the active sequence parameter sets for access unit n and access unit n – 1 differ, the HSS selects a value SchedSelIdx1 of SchedSelIdx from among the values of SchedSelIdx provided in the active sequence parameter set for access unit n that results in a BitRate[SchedSelIdx1] or CpbSize[SchedSelIdx1] for access unit n. The value of BitRate[SchedSelIdx1] or CpbSize[SchedSelIdx1] may differ from the value of BitRate[SchedSelIdx0] or CpbSize[SchedSelIdx0] for the value SchedSelIdx0 of SchedSelIdx that was in use for access unit n - 1.

– Otherwise, the HSS continues to operate with the previous values of SchedSelIdx, BitRate[SchedSelIdx] and CpbSize[SchedSelIdx].

When the HSS selects values of BitRate[SchedSelIdx] or CpbSize[SchedSelIdx] that differ from those of the previous access unit, the following applies.

– the variable BitRate[SchedSelIdx] comes into effect at time $t_{ai}(n)$.

– the variable CpbSize[SchedSelIdx] comes into effect as follows.

  – If the new value of CpbSize[SchedSelIdx] exceeds the old CPB size, it comes into effect at time $t_{ai}(n)$,

  – Otherwise, the new value of CpbSize[SchedSelIdx] comes into effect at the time $t_r(n)$.

C.1.2 Timing of coded picture removal

For access unit 0, the nominal removal time of the access unit from the CPB is specified by

$$t_{r,n}(0) = initial\_cpb\_removal\_delay[SchedSelIdx] \div 90000 \quad (C\text{-}7)$$

For the first access unit of a buffering period that does not initialise the HRD, the nominal removal time of the access unit from the CPB is specified by

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * cpb\_removal\_delay(n) \quad (C\text{-}8)$$

where $t_{r,n}(n_b)$ is the nominal removal time of the first access unit of the previous buffering period and cpb_removal_delay(n) is the value of cpb_removal_delay specified in the picture timing SEI message associated with access unit n.

When an access unit n is the first access unit of a buffering period, n_b is set equal to n at the removal time of access unit n.

The nominal removal time $t_{r,n}(n)$ of an access unit n that is not the first access unit of a buffering period is given by

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * \text{cpb\_removal\_delay}(n) \quad \text{(C-9)}$$

where $t_{r,n}(n_b)$ is the nominal removal time of the first access unit of the current buffering period and cpb_removal_delay(n) is the value of cpb_removal_delay specified in the picture timing SEI message associated with access unit n.

The removal time of access unit n is specified as follows.

– If low_delay_hrd_flag is equal to 0 or $t_{r,n}(n) >= t_{af}(n)$, the removal time of access unit n is specified by

$$t_r(n) = t_{r,n}(n) \quad \text{(C-10)}$$

– Otherwise (low_delay_hrd_flag is equal to 1 and $t_{r,n}(n) < t_{af}(n)$), the removal time of access unit n is specified by

$$t_r(n) = t_{r,n}(n) + t_c * \text{Ceil}((t_{af}(n) - t_{r,n}(n)) \div t_c) \quad \text{(C-11)}$$

NOTE – The latter case indicates that the size of access unit n, b(n), is so large that it prevents removal at the nominal removal time.

C.2 Operation of the decoded picture buffer (DPB)

The decoded picture buffer contains frame buffers. When decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, each of the frame buffers may contain a decoded frame, a decoded complementary field pair or a single (non-paired) decoded field that are marked as "used for reference" (reference pictures) or are held for future output (reordered or delayed pictures). When decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G, each frame buffer may contain a decoded frame, a decoded complementary field pair, a single (non-paired) decoded field, a decoded reference base frame, a decoded reference base complementary field pair or a single (non-paired) decoded reference base field that are marked as "used for reference" (reference pictures) or are held for future output (reordered or delayed pictures). Prior to initialisation, the DPB is empty (the DPB fullness is set to zero). The following steps of the subclauses of this subclause all happen instantaneously at $t_r(n)$ and in the sequence listed.

C.2.1 Decoding of gaps in frame_num and storage of "non-existing" frames

If applicable, gaps in frame_num are detected by the decoding process and the generated frames are marked and inserted into the DPB as specified below.

Gaps in frame_num are detected by the decoding process and the generated frames are marked as specified in subclause 8.2.5.2.

After the marking of each generated frame, each picture m marked by the “sliding window” process as “unused for reference” is removed from the DPB when it is also marked as "non-existing" or its DPB output time is less than or equal to the CPB removal time of the current picture n; i.e., $t_{o,dpb}(m) <= t_r(n)$, or it has OutputFlag equal to 0. When a frame or the last field in a frame buffer is removed from the DPB, the DPB fullness is decremented by one. The “non-existing” generated frame is inserted into the DPB and the DPB fullness is incremented by one.

C.2.2 Picture decoding and output

Picture n is decoded. When it has OutputFlag equal to 1, its DPB output time $t_{o,dpb}(n)$ is derived by

$$t_{o,dpb}(n) = t_r(n) + t_c * \text{dpb\_output\_delay}(n) \quad \text{(C-12)}$$

The output of the current picture is specified as follows.

– If OutputFlag is equal to 1 and $t_{o,dpb}(n) = t_r(n)$, the current picture is output.
NOTE – When the current picture is a reference picture it will be stored in the DPB.

– Otherwise, if OutputFlag is equal to 0, the current picture is not output, but it may be stored in the DPB as specified in subclause C.2.4.

– Otherwise (OutputFlag is equal to 1 and $t_{o,dpb}(n) > t_r(n)$), the current picture is output later and will be stored in the DPB (as specified in subclause C.2.4) and is output at time $t_{o,dpb}(n)$ unless indicated not to be output by the decoding or inference of no_output_of_prior_pics_flag equal to 1 at a time that precedes $t_{o,dpb}(n)$.

When output, the picture shall be cropped, using the cropping rectangle specified in the active sequence parameter set for the picture.

When picture n is a picture that is output and is not the last picture of the bitstream that is output, the value of $\Delta t_{o,dpb}(n)$ is defined as:

$$\Delta t_{o,dpb}(n) = t_{o,dpb}(n_n) - t_{o,dpb}(n) \quad \text{(C-13)}$$

where n_n indicates the picture that follows after picture n in output order and has OutputFlag equal to 1.

The decoded picture is temporarily stored (not in the DPB).

C.2.3 Removal of pictures from the DPB before possible insertion of the current picture

The removal of pictures from the DPB before possible insertion of the current picture proceeds as follows.

- If the decoded picture is an IDR picture the following applies.
  - All reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5.1 when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, or as specified in subclause G.8.2.4when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G.
  - When the IDR picture is not the first IDR picture decoded and the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the active sequence parameter set is different from the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the sequence parameter set that was active for the preceding picture, respectively, no_output_of_prior_pics_flag is inferred to be equal to 1 by the HRD, regardless of the actual value of no_output_of_prior_pics_flag.

    NOTE – Decoder implementations should try to handle frame or DPB size changes more gracefully than the HRD in regard to changes in PicWidthInMbs or FrameHeightInMbs.
  - When no_output_of_prior_pics_flag is equal to 1 or is inferred to be equal to 1, all frame buffers in the DPB are emptied without output of the pictures they contain, and DPB fullness is set to 0.
- Otherwise (the decoded picture is not an IDR picture), the following applies.
  - If the slice header of the current picture includes memory_management_control_operation equal to 5, all reference pictures in the DPB are marked as "unused for reference".
  - Otherwise (the slice header of the current picture does not include memory_management_control_operation equal to 5), the decoded reference picture marking process specified in subclause 8.2.5 is invoked when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, or the decoded reference picture marking process specified in subclause G.8.2.4 is invoked when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G.

All pictures m in the DPB, for which all of the following conditions are true, are removed from the DPB.

- picture m is marked as “unused for reference” or picture m is a non-reference picture. When a picture is a reference frame, it is considered to be marked as "unused for reference" only when both of its fields have been marked as "unused for reference".
- picture m is marked as "non-existing" or it has OutputFlag equal to 0 or its DPB output time is less than or equal to the CPB removal time of the current picture n; i.e., $t_{o,dpb}(m) <= t_r(n)$

When a frame or the last field in a frame buffer is removed from the DPB, the DPB fullness is decremented by one.

C.2.4 Current decoded picture marking and storage

C.2.4.1 Marking and storage of a reference decoded picture into the DPB

When the current picture is a reference picture it is stored in the DPB as follows.

- If the current decoded picture is a second field (in decoding order) of a complementary reference field pair, and the first field of the pair is still in the DPB, the current decoded picture is stored in the same frame buffer as the first field of the pair.

– Otherwise, the current decoded picture is stored in an empty frame buffer, and the DPB fullness is incremented by one.

C.2.4.2 Storage of a non-reference picture into the DPB

When the current picture is a non-reference picture and current picture n has $t_{o,dpb}(n) > t_r(n)$, it is stored in the DPB as follows.

– If the current decoded picture is a second field (in decoding order) of a complementary non-reference field pair, and the first field of the pair is still in the DPB, the current decoded picture is stored in the same frame buffer as the first field of the pair.

– Otherwise, the current decoded picture is stored in an empty frame buffer, and the DPB fullness is incremented by one.

C.3 Bitstream conformance

A bitstream of coded data conforming to this Recommendation | International Standard fulfils the following requirements.

The bitstream is constructed according to the syntax, semantics, and constraints specified in this Recommendation | International Standard outside of this annex.

The bitstream is tested by the HRD as specified below:

For Type I bitstreams, the number of tests carried out is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is either the syntax element of hrd_parameters( ) following the vcl_hrd_parameters_present_flag or is determined by the application by other means not specified in this Recommendation | International Standard. One test is carried out for each bit rate and CPB size combination specified by hrd_parameters( ) following the vcl_hrd_parameters_present_flag. Each of these tests is conducted at the Type I conformance point shown in Figure C-1.

For Type II bitstreams there are two sets of tests. The number of tests of the first set is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is either the syntax element of hrd_parameters( ) following the vcl_hrd_parameters_present_flag or is determined by the application by other means not specified in this Recommendation | International Standard. One test is carried out for each bit rate and CPB size combination. Each of these tests is conducted at the Type I conformance point shown in Figure C-1. For these tests, only VCL and filler data NAL units are counted for the input bit rate and CPB storage.

The number of tests of the second set, for Type II bitstreams, is equal to cpb_cnt_minus1 + 1 where cpb_cnt_minus1 is either the syntax element of hrd_parameters( ) following the nal_hrd_parameters_present_flag or is determined by the application by other means not specified in this Recommendation | International Standard. One test is carried out for each bit rate and CPB size combination specified by hrd_parameters( ) following the nal_hrd_parameters_present_flag. Each of these tests is conducted at the Type II conformance point shown in Figure C-1. For these tests, all NAL units (of a Type II NAL unit stream) or all bytes (of a byte stream) are counted for the input bit rate and CPB storage.

NOTE 1 – NAL HRD parameters established by a value of SchedSelIdx for the Type II conformance point shown in Figure C-1 are sufficient to also establish VCL HRD conformance for the Type I conformance point shown in Figure C-1 for the same values of initial_cpb_removal_delay[SchedSelIdx], BitRate[SchedSelIdx], and CpbSize[SchedSelIdx] for the VBR case (cbr_flag[SchedSelIdx] equal to 0). This is because the data flow into the Type I conformance point is a subset of the data flow into the Type II conformance point and because, for the VBR case, the CPB is allowed to become empty and stay empty until the time a next picture is scheduled to begin to arrive. For example, when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, when NAL HRD parameters are provided for the Type II conformance point that not only fall within the bounds set for NAL HRD parameters for profile conformance in item j of subclause A.3.1 or item i of subclause A.3.3 (depending on the profile in use) but also fall within the bounds set for VCL HRD parameters for profile conformance in item i of subclause A.3.1 or item h of subclause A.3.3 (depending on the profile in use), conformance of the VCL HRD for the Type I conformance point is also assured to fall within the bounds of item i of subclause A.3.1.

For conforming bitstreams, all of the following conditions shall be fulfilled for each of the tests.

– For each access unit n, with n>0, associated with a buffering period SEI message, with $\Delta t_{g,90}(n)$ specified by

$$\Delta t_{g,90}(n) = 90000 * (t_{r,n}(n) - t_{af}(n - 1)) \quad \text{(C-14)}$$

The value of initial_cpb_removal_delay[SchedSelIdx] shall be constrained as follows.

– If cbr_flag[SchedSelIdx] is equal to 0,

$$\text{initial\_cpb\_removal\_delay[SchedSelIdx]} <= \text{Ceil}(\Delta t_{g,90}(n)) \quad \text{(C-15)}$$

– Otherwise (cbr_flag[SchedSelIdx] is equal to 1),

$$\text{Floor}(\Delta t_{g,90}(n)) <= \text{initial\_cpb\_removal\_delay}[\text{SchedSelIdx}] <= \text{Ceil}(\Delta t_{g,90}(n)) \quad \text{(C-16)}$$

NOTE 2 – The exact number of bits in the CPB at the removal time of each picture may depend on which buffering period SEI message is selected to initialise the HRD. Encoders must take this into account to ensure that all specified constraints must be obeyed regardless of which buffering period SEI message is selected to initialise the HRD, as the HRD may be initialised at any one of the buffering period SEI messages.

– A CPB overflow is specified as the condition in which the total number of bits in the CPB is larger than the CPB size. The CPB shall never overflow.

– A CPB underflow is specified as the condition in which tr,n(n) is less than taf(n). When low_delay_hrd_flag is equal to 0, the CPB shall never underflow.

– The nominal removal times of pictures from the CPB (starting from the second picture in decoding order), shall satisfy the constraints on $t_{r,n}(n)$ and $t_r(n)$ expressed in subclauses A.3.1 through A.3.3 for the profile and level specified in the bitstream when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, and they shall satisfy the constraints on tr,n(n) and tr(n) expressed in subclauses G.10.2.1 and G.10.2.2 for profile and level specified in the bitstream when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G.

– Immediately after any decoded picture is added to the DPB, the fullness of the DPB shall be less than or equal to the DPB size as constrained by Annexes A, D, and E for the profile and level specified in the bitstream.

– All reference pictures shall be present in the DPB when needed for prediction. Each picture shall be present in the DPB at its DPB output time unless it is not stored in the DPB at all, or is removed from the DPB before its output time by one of the processes specified in subclause C.2.

– The value of $\Delta_{to,dpb}(n)$ as given by Equation C-13, which is the difference between the output time of a picture and that of the first picture following it in output order and having OutputFlag equal to 1, shall satisfy the constraint expressed in subclause A.3.1 for the profile and level specified in the bitstream when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, and it shall satisfy the constraint expressed in subclause G.10.2.1 for profile and level specified in the bitstream when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G.

C.4 Decoder conformance

A decoder conforming to this Recommendation | International Standard fulfils the following requirements.

A decoder claiming conformance to a specific profile and level shall be able to decode successfully all conforming bitstreams specified for decoder conformance in subclause C.3, provided that all sequence parameter sets and picture parameters sets referred to in the VCL NAL units, and appropriate buffering period and picture timing SEI messages are conveyed to the decoder, in a timely manner, either in the bitstream (by non-VCL NAL units), or by external means not specified by this Recommendation | International Standard.

There are two types of conformance that can be claimed by a decoder: output timing conformance and output order conformance.

To check conformance of a decoder, test bitstreams conforming to the claimed profile and level, as specified by subclause C.3 are delivered by a hypothetical stream scheduler (HSS) both to the HRD and to the decoder under test (DUT). All pictures output by the HRD shall also be output by the DUT and, for each picture output by the HRD, the values of all samples that are output by the DUT for the corresponding picture shall be equal to the values of the samples output by the HRD.

For output timing decoder conformance, the HSS operates as described above, with delivery schedules selected only from the subset of values of SchedSelIdx for which the bit rate and CPB size are restricted as specified in Annex A for the specified profile and level, or with "interpolated" delivery schedules as specified below for which the bit rate and CPB size are restricted as specified in Annex A. The same delivery schedule is used for both the HRD and DUT.

When the HRD parameters and the buffering period SEI messages are present with cpb_cnt_minus1 greater than 0, the decoder shall be capable of decoding the bitstream as delivered from the HSS operating using an "interpolated" delivery schedule specified as having peak bit rate r, CPB size c(r), and initial CPB removal delay (f(r) ÷ r) as follows

$$\alpha = (r - \text{BitRate}[\text{SchedSelIdx} - 1]) \div (\text{BitRate}[\text{SchedSelIdx}] - \text{BitRate}[\text{SchedSelIdx} - 1]), \quad \text{(C-17)}$$

$$c(r) = \alpha * CpbSize[SchedSelIdx] + (1 - \alpha) * CpbSize[SchedSelIdx-1], \quad \text{(C-18)}$$

$$f(r) = \alpha * initial\_cpb\_removal\_delay[SchedSelIdx] * BitRate[SchedSelIdx] + (1 - \alpha) * initial\_cpb\_removal\_delay[SchedSelIdx - 1] * BitRate[SchedSelIdx - 1] \quad \text{(C-19)}$$

for any SchedSelIdx > 0 and r such that BitRate[SchedSelIdx - 1] <= r <= BitRate[SchedSelIdx] such that r and c(r) are within the limits as specified in Annex A for the maximum bit rate and buffer size for the specified profile and level.

NOTE 1 – initial_cpb_removal_delay[SchedSelIdx] can be different from one buffering period to another and have to be re-calculated.

For output timing decoder conformance, an HRD as described above is used and the timing (relative to the delivery time of the first bit) of picture output is the same for both HRD and the DUT up to a fixed delay.

For output order decoder conformance, the HSS delivers the bitstream to the DUT "by demand" from the DUT, meaning that the HSS delivers bits (in decoding order) only when the DUT requires more bits to proceed with its processing.

NOTE 2 – This means that for this test, the coded picture buffer of the DUT could be as small as the size of the largest access unit.

A modified HRD as described below is used, and the HSS delivers the bitstream to the HRD by one of the schedules specified in the bitstream such that the bit rate and CPB size are restricted as specified in Annex A. The order of pictures output shall be the same for both HRD and the DUT.

For output order decoder conformance, the HRD CPB size is equal to CpbSize[SchedSelIdx] for the selected schedule and the DPB size is equal to MaxDpbSize. Removal time from the CPB for the HRD is equal to final bit arrival time and decoding is immediate. The operation of the DPB of this HRD is described below.

C.4.1 Operation of the output order DPB

The decoded picture buffer contains frame buffers. When decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, each of the frame buffers may contain a decoded frame, a decoded complementary field pair or a single (non-paired) decoded field that is marked as "used for reference" or is held for future output (reordered pictures). When decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G, each frame buffer may contain a decoded frame, a decoded complementary field pair, a single (non-paired) decoded field, a decoded reference base frame, a decoded reference base complementary field pair or a single (non-paired) decoded reference base field that are marked as "used for reference" (reference pictures) or are held for future output (reordered or delayed pictures). At HRD initialisation, the DPB fullness, measured in frames, is set to 0. The following steps all happen instantaneously when an access unit is removed from the CPB, and in the order listed.

C.4.2 Decoding of gaps in frame_num and storage of "non-existing" pictures

When applicable, gaps in frame_num are detected by the decoding process and the necessary number of "non-existing" frames are inferred in the order specified by the generation of values of UnusedShortTermFrameNum in Equation 7-22 and are marked as specified in subclause 8.2.5.2. Frame buffers containing a frame or a complementary field pair or a non-paired field which are marked as "not needed for output" and "unused for reference" are emptied (without output), and the DPB fullness is decremented by the number of frame buffers emptied. Each "non-existing" frame is stored in the DPB as follows.

- When there is no empty frame buffer (i.e., DPB fullness is equal to DPB size), the "bumping" process specified in subclause C.4.5.3 is invoked repeatedly until there is an empty frame buffer in which to store the "non-existing" frame.
- The "non-existing" frame is stored in an empty frame buffer and is marked as "not needed for output", and the DPB fullness is incremented by one.

C.4.3 Picture decoding

Primary coded picture n is decoded and is temporarily stored (not in the DPB).

C.4.4 Removal of pictures from the DPB before possible insertion of the current picture

The removal of pictures from the DPB before possible insertion of the current picture proceeds as follows.

- If the decoded picture is an IDR picture the following applies.
  - All reference pictures in the DPB are marked as "unused for reference" as specified in subclause 8.2.5 when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, or as specified in subclause G.8.2.4 when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G.
  - When the IDR picture is not the first IDR picture decoded and the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the active sequence parameter set is different from the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering derived from the sequence parameter set that was active for the preceding picture, respectively, no_output_of_prior_pics_flag is inferred to be equal to 1 by the HRD, regardless of the actual value of no_output_of_prior_pics_flag.

    NOTE – Decoder implementations should try to handle changes in the value of PicWidthInMbs or FrameHeightInMbs or max_dec_frame_buffering more gracefully than the HRD.
  - When no_output_of_prior_pics_flag is equal to 1 or is inferred to be equal to 1, all frame buffers in the DPB are emptied without output of the pictures they contain, and DPB fullness is set to 0.
- Otherwise (the decoded picture is not an IDR picture), the decoded reference picture marking process is invoked as specified in subclause 8.2.5 when decoding a coded video sequence conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2-9, or as specified in subclause G.8.2.4 when decoding a coded video sequence conforming to one or more of the profiles specified in Annex G using the decoding process specified in Annex G. Frame buffers containing a frame or a complementary field pair or a non-paired field which are marked as "not needed for output" and "unused for reference" are emptied (without output), and the DPB fullness is decremented by the number of frame buffers emptied.

When the current picture has a memory_management_control_operation equal to 5 or is an IDR picture for which no_output_of_prior_pics_flag is not equal to 1 and is not inferred to be equal to 1, the following two steps are performed.

1. Frame buffers containing a frame or a complementary field pair or a non-paired field which are marked as "not needed for output" and "unused for reference" are emptied (without output), and the DPB fullness is decremented by the number of frame buffers emptied.
2. All non-empty frame buffers in the DPB are emptied by repeatedly invoking the “bumping” process specified in subclause C.4.5.3, and the DPB fullness is set to 0.

C.4.5 Current decoded picture marking and storage

C.4.5.1 Storage and marking of a reference decoded picture into the DPB

When the current picture is a reference picture, it is stored in the DPB as follows.

- If the current decoded picture is the second field (in decoding order) of a complementary reference field pair, and the first field of the pair is still in the DPB, the current picture is stored in the same frame buffer as the first field of the pair and the following applies.
  - If the current decoded picture has OutputFlag equal to 1, it is marked as "needed for output".
  - Otherwise (the current decoded picture has OutputFlag equal to 0), it is marked as "not needed for output".
- Otherwise, the following operations are performed:
  - When there is no empty frame buffer (i.e., DPB fullness is equal to DPB size), the "bumping" process specified in subclause C.4.5.3 is invoked repeatedly until there is an empty frame buffer in which to store the current decoded picture.
  - The current decoded picture is stored in an empty frame buffer, the DPB fullness is incremented by one, and the following applies.
    - If the current decoded picture has OutputFlag equal to 1, it is marked as “needed for output”.
    - Otherwise (the current decoded picture has OutputFlag equal to 0), it is marked as “not needed for output”.

C.4.5.2 Storage and marking of a non-reference decoded picture into the DPB

When the current picture is a non-reference picture, the following operations are performed.

- If the current decoded picture is the second field (in decoding order) of a complementary non-reference field pair and the first field of the pair is still in the DPB, the current picture is stored in the same frame buffer as the first field of the pair and the following applies.
  - If the current decoded picture has OutputFlag equal to 1, it is marked as "needed for output".
  - Otherwise (the current decoded picture has OutputFlag equal to 0), it is marked as "not needed for output".
- Otherwise, if the current picture has OutputFlag equal to 0, the DPB is not modified and the current picture is not output.
- Otherwise, the following operations are performed repeatedly until the current decoded picture has been cropped and output or has been stored in the DPB:
  - If there is no empty frame buffer (i.e., DPB fullness is equal to DPB size), the following applies.
    - If the current picture does not have a lower value of PicOrderCnt( ) than all pictures in the DPB that are marked as "needed for output", the "bumping" process described in subclause C.4.5.3 is performed.
    - Otherwise (the current picture has a lower value of PicOrderCnt( ) than all pictures in the DPB that are marked as "needed for output"), the current picture is cropped, using the cropping rectangle specified in the active sequence parameter set for the picture and the cropped picture is output.
  - Otherwise (there is an empty frame buffer, i.e., DPB fullness is less than DPB size) the current decoded picture is stored in an empty frame buffer and is marked as "needed for output", and the DPB fullness is incremented by one.

C.4.5.3 "Bumping" process

The "bumping" process is invoked in the following cases.

- There is no empty frame buffer (i.e., DPB fullness is equal to DPB size) and an empty frame buffer is needed for storage of an inferred "non-existing" frame, as specified in subclause C.4.2.
- The current picture is an IDR picture and no_output_of_prior_pics_flag is not equal to 1 and is not inferred to be equal to 1, as specified in subclause C.4.4.
- The current picture has memory_management_control_operation equal to 5, as specified in subclause C.4.4.
- There is no empty frame buffer (i.e., DPB fullness is equal to DPB size) and an empty frame buffer is needed for storage of a decoded (non-IDR) reference picture, as specified in subclause C.4.5.1.
- There is no empty frame buffer (i.e., DPB fullness is equal to DPB size) and the current picture is a non-reference picture that is not the second field of a complementary non-reference field pair and the current picture has OutputFlag equal to 1 and there are pictures in the DPB that are marked as "needed for output" that precede the current non-reference picture in output order, as specified in subclause C.4.5.2, so an empty buffer is needed for storage of the current picture.

The "bumping" process consists of the following:

- The picture or complementary reference field pair that is first for output is selected as follows.
  - The frame buffer is selected that contains the picture having the smallest value of PicOrderCnt( ) of all pictures in the DPB marked as "needed for output".
  - If this frame buffer contains a complementary non-reference field pair with both fields marked as "needed for output" and both fields have the same PicOrderCnt( ), the first of these two fields in decoding order is considered first for output.
  - Otherwise, if this frame buffer contains a complementary reference field pair with both fields marked as "needed for output" and both fields have the same PicOrderCnt( ), the entire complementary reference field pair is considered first for output.
  - Otherwise, the picture in this frame buffer that has the smallest value of PicOrderCnt( ) is considered first for output.
- If a single picture is considered first for output, this picture is cropped, using the cropping rectangle specified in the active sequence parameter set for the picture, the cropped picture is output, and the picture is marked as "not needed for output".

- Otherwise (a complementary reference field pair is considered first for output), the two fields of the complementary reference field pair are both cropped, using the cropping rectangle specified in the active sequence parameter set for the pictures, the two fields of the complementary reference field pair are output together, and both fields of the complementary reference field pair are marked as "not needed for output".
- The frame buffer that included the picture or complementary reference field pair that was cropped and output is checked, and when any of the following conditions is satisfied, the frame buffer is emptied and the DPB fullness is decremented by 1.
  - The frame buffer contains a non-reference non-paired field.
  - The frame buffer contains a non-reference frame.
  - The frame buffer contains a complementary non-reference field pair with both fields marked as "not needed for output".
  - The frame buffer contains a non-paired reference field marked as "unused for reference".
  - The frame buffer contains a reference frame with both fields marked as "unused for reference".
  - The frame buffer contains a complementary reference field pair with both fields marked as "unused for reference" and "not needed for output".

Annex D
Supplemental enhancement information
(This annex forms an integral part of this Recommendation | International Standard)

This annex specifies syntax and semantics for SEI message payloads.

SEI messages assist in processes related to decoding, display or other purposes. However, SEI messages are not required for constructing the luma or chroma samples by the decoding process. Conforming decoders are not required to process this information for output order conformance to this Recommendation | International Standard (see Annex C for the specification of conformance). Some SEI message information is required to check bitstream conformance and for output timing decoder conformance.

In Annex D, specification for presence of SEI messages are also satisfied when those messages (or some subset of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. When present in the bitstream, SEI messages shall obey the syntax and semantics specified in subclauses 7.3.2.3 and 7.4.2.3 and this annex. When the content of an SEI message is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the SEI message is not required to use the same syntax specified in this annex. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

D.1 SEI payload syntax

sei_payload(payloadType, payloadSize) {	C	Descriptor
if(payloadType == 0)		
buffering_period(payloadSize)	5	
else if(payloadType == 1)		
pic_timing(payloadSize)	5	
else if(payloadType == 2)		
pan_scan_rect(payloadSize)	5	
else if(payloadType == 3)		
filler_payload(payloadSize)	5	
else if(payloadType == 4)		
user_data_registered_itu_t_t35(payloadSize)	5	
else if(payloadType == 5)		
user_data_unregistered(payloadSize)	5	
else if(payloadType == 6)		
recovery_point(payloadSize)	5	
else if(payloadType == 7)		
dec_ref_pic_marking_repetition(payloadSize)	5	
else if(payloadType == 8)		
spare_pic(payloadSize)	5	
else if(payloadType == 9)		
scene_info(payloadSize)	5	
else if(payloadType == 10)		
sub_seq_info(payloadSize)	5	
else if(payloadType == 11)		
sub_seq_layer_characteristics(payloadSize)	5	
else if(payloadType == 12)		
sub_seq_characteristics(payloadSize)	5	

else if(payloadType == 13)		
full_frame_freeze(payloadSize)	5	
else if(payloadType == 14)		
full_frame_freeze_release(payloadSize)	5	
else if(payloadType == 15)		
full_frame_snapshot(payloadSize)	5	
else if(payloadType == 16)		
progressive_refinement_segment_start(payloadSize)	5	
else if(payloadType == 17)		
progressive_refinement_segment_end(payloadSize)	5	
else if(payloadType == 18)		
motion_constrained_slice_group_set(payloadSize)	5	
else if(payloadType == 19)		
film_grain_characteristics(payloadSize)	5	
else if(payloadType == 20)		
deblocking_filter_display_preference(payloadSize)	5	
else if(payloadType == 21)		
stereo_video_info(payloadSize)	5	
else if(payloadType == 22)		
post_filter_hint(payloadSize)	5	
else if(payloadType == 23)		
tone_mapping_info(payloadSize)	5	
else if(payloadType == 24)		
scalability_info(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 25)		
sub_pic_scalable_layer(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 26)		
non_required_layer_rep(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 27)		
priority_layer_info(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 28)		
layers_not_present(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 29)		
layer_dependency_change(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 30)		
scalable_nesting(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 31)		
base_layer_temporal_hrd(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 32)		
quality_layer_integrity_check(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 33)		
redundant_pic_property(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 34)		
tl0_picture_index(payloadSize) /* specified in Annex G */	5	
else if(payloadType == 35)		
tl_switching_point(payloadSize) /* specified in Annex G */	5	
else		

reserved_sei_message(payloadSize)	5	
if(!byte_aligned( )) {		
bit_equal_to_one /* equal to 1 */	5	f(1)
while(!byte_aligned( ))		
bit_equal_to_zero /* equal to 0 */	5	f(1)
}		
}		

D.1.1 Buffering period SEI message syntax

buffering_period(payloadSize) {	C	Descriptor
seq_parameter_set_id	5	ue(v)
if(NalHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
if(VclHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
}		

D.1.2 Picture timing SEI message syntax

pic_timing(payloadSize) {	C	Descriptor
if(CpbDpbDelaysPresentFlag) {		
cpb_removal_delay	5	u(v)
dpb_output_delay	5	u(v)
}		
if(pic_struct_present_flag) {		
pic_struct	5	u(4)
for(i = 0; i < NumClockTS ; i++) {		
clock_timestamp_flag[i]	5	u(1)
if(clock_timestamp_flag[i]) {		
ct_type	5	u(2)
nuit_field_based_flag	5	u(1)
counting_type	5	u(5)
full_timestamp_flag	5	u(1)
discontinuity_flag	5	u(1)
cnt_dropped_flag	5	u(1)
n_frames	5	u(8)
if(full_timestamp_flag) {		

seconds_value /* 0..59 */	5	u(6)
minutes_value /* 0..59 */	5	u(6)
hours_value /* 0..23 */	5	u(5)
} else {		
seconds_flag	5	u(1)
if(seconds_flag) {		
seconds_value /* range 0..59 */	5	u(6)
minutes_flag	5	u(1)
if(minutes_flag) {		
minutes_value /* 0..59 */	5	u(6)
hours_flag	5	u(1)
if(hours_flag)		
hours_value /* 0..23 */	5	u(5)
}		
}		
}		
if(time_offset_length > 0)		
time_offset	5	i(v)
}		
}		
}		
}		

D.1.3 Pan-scan rectangle SEI message syntax

pan_scan_rect(payloadSize) {	C	Descriptor
pan_scan_rect_id	5	ue(v)
pan_scan_rect_cancel_flag	5	u(1)
if(!pan_scan_rect_cancel_flag) {		
pan_scan_cnt_minus1	5	ue(v)
for(i = 0; i <= pan_scan_cnt_minus1; i++) {		
pan_scan_rect_left_offset[i]	5	se(v)
pan_scan_rect_right_offset[i]	5	se(v)
pan_scan_rect_top_offset[i]	5	se(v)
pan_scan_rect_bottom_offset[i]	5	se(v)
}		
pan_scan_rect_repetition_period	5	ue(v)
}		
}		

D.1.4 Filler payload SEI message syntax

filler_payload(payloadSize) {	C	Descriptor
for(k = 0; k < payloadSize; k++)		
ff_byte /* equal to 0xFF */	5	f(8)
}		

D.1.5 User data registered by ITU-T Rec. T.35 SEI message syntax

user_data_registered_itu_t_t35(payloadSize) {	C	Descriptor
itu_t_t35_country_code	5	b(8)
if(itu_t_t35_country_code != 0xFF)		
i = 1		
else {		
itu_t_t35_country_code_extension_byte	5	b(8)
i = 2		
}		
do {		
itu_t_t35_payload_byte	5	b(8)
i++		
} while(i < payloadSize)		
}		

D.1.6 User data unregistered SEI message syntax

user_data_unregistered(payloadSize) {	C	Descriptor
uuid_iso_iec_11578	**5**	u(128)
for(i = 16; i < payloadSize; i++)		
user_data_payload_byte	5	b(8)
}		

D.1.7 Recovery point SEI message syntax

recovery_point(payloadSize) {	C	Descriptor
recovery_frame_cnt	5	ue(v)
exact_match_flag	5	u(1)
broken_link_flag	5	u(1)
changing_slice_group_idc	5	u(2)
}		

D.1.8 Decoded reference picture marking repetition SEI message syntax

dec_ref_pic_marking_repetition(payloadSize) {	C	Descriptor
original_idr_flag	5	u(1)
original_frame_num	5	ue(v)
if(!frame_mbs_only_flag) {		
original_field_pic_flag	5	u(1)
if(original_field_pic_flag)		
original_bottom_field_flag	5	u(1)
}		
dec_ref_pic_marking( )	5	
}		

D.1.9 Spare picture SEI message syntax

spare_pic(payloadSize) {	C	Descriptor
target_frame_num	5	ue(v)
spare_field_flag	5	u(1)
if(spare_field_flag)		
target_bottom_field_flag	5	u(1)
num_spare_pics_minus1	5	ue(v)
for(i = 0; i < num_spare_pics_minus1 + 1; i++) {		
delta_spare_frame_num[i]	5	ue(v)
if(spare_field_flag)		
spare_bottom_field_flag[i]	5	u(1)
spare_area_idc[i]	5	ue(v)
if(spare_area_idc[i] == 1)		
for(j = 0; j < PicSizeInMapUnits; j++)		
spare_unit_flag[i][j]	5	u(1)
else if(spare_area_idc[i] == 2) {		
mapUnitCnt = 0		
for(j=0; mapUnitCnt < PicSizeInMapUnits; j++) {		
zero_run_length[i][j]	5	ue(v)
mapUnitCnt += zero_run_length[i][j] + 1		
}		
}		
}		
}		

D.1.10 Scene information SEI message syntax

scene_info(payloadSize) {	C	Descriptor
scene_info_present_flag	5	u(1)
if(scene_info_present_flag) {		
scene_id	5	ue(v)
scene_transition_type	5	ue(v)
if(scene_transition_type > 3)		
second_scene_id	5	ue(v)
}		
}		

D.1.11 Sub-sequence information SEI message syntax

sub_seq_info(payloadSize) {	C	Descriptor
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
first_ref_pic_flag	5	u(1)
leading_non_ref_pic_flag	5	u(1)
last_pic_flag	5	u(1)
sub_seq_frame_num_flag	5	u(1)
if(sub_seq_frame_num_flag)		
sub_seq_frame_num	5	ue(v)
}		

D.1.12 Sub-sequence layer characteristics SEI message syntax

sub_seq_layer_characteristics(payloadSize) {	C	Descriptor
num_sub_seq_layers_minus1	5	ue(v)
for(layer = 0; layer <= num_sub_seq_layers_minus1; layer++) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
}		

D.1.13 Sub-sequence characteristics SEI message syntax

sub_seq_characteristics(payloadSize) {	C	Descriptor
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
duration_flag	5	u(1)
if(duration_flag)		
sub_seq_duration	5	u(32)
average_rate_flag	5	u(1)
if(average_rate_flag) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
num_referenced_subseqs	5	ue(v)
for(n = 0; n < num_referenced_subseqs; n++) {		
ref_sub_seq_layer_num	5	ue(v)
ref_sub_seq_id	5	ue(v)
ref_sub_seq_direction	5	u(1)
}		
}		

D.1.14 Full-frame freeze SEI message syntax

full_frame_freeze(payloadSize) {	C	Descriptor
full_frame_freeze_repetition_period	5	ue(v)
}		

D.1.15 Full-frame freeze release SEI message syntax

full_frame_freeze_release(payloadSize) {	C	Descriptor
}		

D.1.16 Full-frame snapshot SEI message syntax

full_frame_snapshot(payloadSize) {	C	Descriptor
snapshot_id	5	ue(v)
}		

D.1.17 Progressive refinement segment start SEI message syntax

progressive_refinement_segment_start(payloadSize) {	C	Descriptor
progressive_refinement_id	5	ue(v)
num_refinement_steps_minus1	5	uc(v)
}		

D.1.18 Progressive refinement segment end SEI message syntax

progressive_refinement_segment_end(payloadSize) {	C	Descriptor
progressive_refinement_id	5	ue(v)
}		

D.1.19 Motion-constrained slice group set SEI message syntax

motion_constrained_slice_group_set(payloadSize) {	C	Descriptor
num_slice_groups_in_set_minus1	5	ue(v)
for(i = 0; i <= num_slice_groups_in_set_minus1; i++)		
slice_group_id[i]	5	u(v)
exact_sample_value_match_flag	5	u(1)
pan_scan_rect_flag	5	u(1)
if(pan_scan_rect_flag)		
pan_scan_rect_id	5	ue(v)
}		

D.1.20 Film grain characteristics SEI message syntax

film_grain_characteristics(payloadSize) {	C	Descriptor
film_grain_characteristics_cancel_flag	5	u(1)
if(!film_grain_**characteristics**_cancel_flag) {		
model_id	5	u(2)
separate_colour_description_present_flag	5	u(1)
if(separate_colour_description_present_flag) {		
film_grain_bit_depth_luma_minus8	5	u(3)
film_grain_bit_depth_chroma_minus8	5	u(3)
film_grain_full_range_flag	5	u(1)
film_grain_colour_primaries	5	u(8)
film_grain_transfer_characteristics	5	u(8)
film_grain_matrix_coefficients	5	u(8)
}		
blending_mode_id	5	u(2)
log2_scale_factor	5	u(4)
for(c = 0; c < 3; c++)		
comp_model_present_flag[c **]**	5	u(1)
for(c = 0; c < 3; c++)		
if(comp_model_present_flag[c]) {		
num_intensity_intervals_minus1[c **]**	5	u(8)
num_model_values_minus1[c **]**	5	u(3)
for(i = 0; i <= num_intensity_intervals_minus1[c]; i++) {		
intensity_interval_lower_bound[c **][** i **]**	5	u(8)
intensity_interval_upper_bound[c **][** i **]**	5	u(8)
for(j = 0; j <= num_model_values_minus1[c]; j++)		
comp_model_value[c **][** i **][** j **]**	5	se(v)
}		
}		
film_grain_characteristics_repetition_period	5	ue(v)
}		
}		

D.1.21 Deblocking filter display preference SEI message syntax

deblocking_filter_display_preference(payloadSize) {	C	Descriptor
deblocking_display_preference_cancel_flag	5	u(1)
if(!deblocking_display_preference_cancel_flag) {		
display_prior_to_deblocking_preferred_flag	5	u(1)
dec_frame_buffering_constraint_flag	5	u(1)
deblocking_display_preference_repetition_period	5	ue(v)
}		
}		

D.1.22 Stereo video information SEI message syntax

stereo_video_info(payloadSize) {	C	Descriptor
field_views_flag	5	u(1)
if(field_views_flag)		
top_field_is_left_view_flag	5	u(1)
else {		
current_frame_is_left_view_flag	5	u(1)
next_frame_is_second_view_flag	5	u(1)
}		
left_view_self_contained_flag	5	u(1)
right_view_self_contained_flag	5	u(1)
}		

D.1.23 Post-filter hint SEI message syntax

post_filter_hint(payloadSize) {	C	Descriptor
filter_hint_size_y	5	ue(v)
filter_hint_size_x	5	ue(v)
filter_hint_type	5	u(2)
for(colour_component = 0; colour_component < 3; colour_component ++)		
for(cy = 0; cy < filter_hint_size_y; cy ++)		
for(cx = 0; cx < filter_hint_size_x; cx ++)		
filter_hint[colour_component **][** cy **][** cx **]**	5	se(v)
additional_extension_flag	5	u(1)
}		

D.1.24 Tone mapping information SEI message syntax

tone_mapping_info(payloadSize) {	C	Descriptor
tone_map_id	5	ue(v)
tone_map_cancel_flag	5	u(1)
if(!tone_map_cancel_flag) {		
tone_map_repetition_period	5	ue(v)
coded_data_bit_depth	5	u(8)
target_bit_depth	5	u(8)
model_id	5	ue(v)
if(model_id == 0) {		
min_value	5	u(32)
max_value	5	u(32)
}		
if(model_id == 1) {		
sigmoid_midpoint	5	u(32)
sigmoid_width	5	u(32)
}		
if(model_id == 2)		
for(i = 0; i < (1 << target_bit_depth); i++)		
start_of_coded_interval[i **]**	5	u(v)
if(model_id == 3) {		
num_pivots	5	u(16)
for(i=0; i < num_pivots; i++) {		
coded_pivot_value[i **]**	5	u(v)
target_pivot_value[i **]**	5	u(v)
}		
}		
}		
}		

D.1.25 Reserved SEI message syntax

reserved_sei_message(payloadSize) {	C	Descriptor
for(i = 0; i < payloadSize; i++)		
reserved_sei_message_payload_byte	5	b(8)
}		

D.2 SEI payload semantics

D.2.1 Buffering period SEI message semantics

When NalHrdBpPresentFlag or VclHrdBpPresentFlag are equal to 1, a buffering period SEI message can be associated with any access unit in the bitstream, and a buffering period SEI message shall be associated with each IDR access unit and with each access unit associated with a recovery point SEI message.

NOTE – For some applications, the frequent presence of a buffering period SEI message may be desirable.

A buffering period is specified as the set of access units between two instances of the buffering period SEI message in decoding order.

seq_parameter_set_id specifies the sequence parameter set that contains the sequence HRD attributes. The value of seq_parameter_set_id shall be equal to the value of seq_parameter_set_id in the picture parameter set referenced by the primary coded picture associated with the buffering period SEI message. The value of seq_parameter_set_id shall be in the range of 0 to 31, inclusive.

initial_cpb_removal_delay[SchedSelIdx] specifies the delay for the SchedSelIdx-th CPB between the time of arrival in the CPB of the first bit of the coded data associated with the access unit associated with the buffering period SEI message and the time of removal from the CPB of the coded data associated with the same access unit, for the first buffering period after HRD initialisation. The syntax element has a length in bits given by initial_cpb_removal_delay_length_minus1 + 1. It is in units of a 90 kHz clock. initial_cpb_removal_delay[SchedSelIdx] shall not be equal to 0 and shall not exceed 90000 * (CpbSize[SchedSelIdx] ÷ BitRate[SchedSelIdx]), the time-equivalent of the CPB size in 90 kHz clock units.

initial cpb removal delay offset[SchedSelIdx] is used for the SchedSelIdx-th CPB in combination with the cpb_removal_delay to specify the initial delivery time of coded access units to the CPB. initial_cpb_removal_delay_offset[SchedSelIdx] is in units of a 90 kHz clock. The initial_cpb_removal_delay_offset[SchedSelIdx] syntax element is a fixed length code whose length in bits is given by initial_cpb_removal_delay_length_minus1 + 1. This syntax element is not used by decoders and is needed only for the delivery scheduler (HSS) specified in Annex C.

Over the entire coded video sequence, the sum of initial_cpb_removal_delay[SchedSelIdx] and initial_cpb_removal_delay_offset[SchedSelIdx] shall be constant for each value of SchedSelIdx.

D.2.2 Picture timing SEI message semantics

The presence of picture timing SEI message in the bitstream is specified as follows.

– If CpbDpbDelaysPresentFlag is equal to 1 or pic_struct_present_flag is equal to 1, one picture timing SEI message shall be present in every access unit of the coded video sequence.

– Otherwise (CpbDpbDelaysPresentFlag is equal to 0 and pic_struct_present_flag is equal to 0), no picture timing SEI messages shall be present in any access unit of the coded video sequence.

cpb_removal_delay specifies how many clock ticks (see subclause E.2.1) to wait after removal from the CPB of the access unit associated with the most recent buffering period SEI message before removing from the buffer the access unit data associated with the picture timing SEI message. This value is also used to calculate an earliest possible time of arrival of access unit data into the CPB for the HSS, as specified in Annex C. The syntax element is a fixed length code whose length in bits is given by cpb_removal_delay_length_minus1 + 1. The cpb_removal_delay is the remainder of a $2^{(\text{cpb\_removal\_delay\_length\_minus1} + 1)}$ counter.

The value of cpb_removal_delay for the first picture in the bitstream shall be equal to 0.

dpb_output_delay is used to compute the DPB output time of the picture. It specifies how many clock ticks to wait after removal of an access unit from the CPB before the decoded picture can be output from the DPB (see subclause C.2).

NOTE 1 – A picture is not removed from the DPB at its output time when it is still marked as "used for short-term reference" or "used for long-term reference".

NOTE 2 – Only one dpb_output_delay is specified for a decoded picture.

The size of the syntax element dpb_output_delay is given in bits by dpb_output_delay_length_minus1 + 1. When max_dec_frame_buffering is equal to 0, dpb_output_delay shall be equal to 0.

The output time derived from the dpb_output_delay of any picture that is output from an output timing conforming decoder as specified in subclause C.2 shall precede the output time derived from the dpb_output_delay of all pictures in any subsequent coded video sequence in decoding order.

The output time derived from the dpb_output_delay of the second field, in decoding order, of a complementary non-reference field pair shall exceed the output time derived from the dpb_output_delay of the first field of the same complementary non-reference field pair.

The picture output order established by the values of this syntax element shall be the same order as established by the values of PicOrderCnt( ) as specified by subclauses C.4.1 to C.4.5, except that when the two fields of a complementary reference field pair have the same value of PicOrderCnt( ), the two fields have different output times.

For pictures that are not output by the "bumping" process of subclause C.4.5 because they precede, in decoding order, an IDR picture with no_output_of_prior_pics_flag equal to 1 or inferred to be equal to 1, the output times derived from dpb_output_delay shall be increasing with increasing value of PicOrderCnt( ) relative to all pictures within the same coded video sequence subsequent to any picture having a memory_management_control_operation equal to 5.

pic_struct indicates whether a picture should be displayed as a frame or one or more fields, according to Table D-1. Frame doubling (pic_struct equal to 7) indicates that the frame should be displayed two times consecutively, and frame tripling (pic_struct equal to 8) indicates that the frame should be displayed three times consecutively.

NOTE 3 – Frame doubling can facilitate the display, for example, of 25p video on a 50p display and 29.97p video on a 59.94p display. Using frame doubling and frame tripling in combination on every other frame can facilitate the display of 23.98p video on a 59.94p display.

Table D-1 – Interpretation of pic_struct

Value	Indicated display of picture	Restrictions	NumClockTS
0	frame	field_pic_flag shall be 0	1
1	top field	field_pic_flag shall be 1, bottom_field_flag shall be 0	1
2	bottom field	field_pic_flag shall be 1, bottom_field_flag shall be 1	1
3	top field, bottom field, in that order	field_pic_flag shall be 0	2
4	bottom field, top field, in that order	field_pic_flag shall be 0	2
5	top field, bottom field, top field repeated, in that order	field_pic_flag shall be 0	3
6	bottom field, top field, bottom field repeated, in that order	field_pic_flag shall be 0	3
7	frame doubling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	2
8	frame tripling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	3
9..15	reserved		

NumClockTS is determined by pic_struct as specified in Table D-1. There are up to NumClockTS sets of clock timestamp information for a picture, as specified by clock_timestamp_flag[i] for each set. The sets of clock timestamp information apply to the field(s) or the frame(s) associated with the picture by pic_struct.

The contents of the clock timestamp syntax elements indicate a time of origin, capture, or alternative ideal display. This indicated time is computed as

$$\begin{aligned} \text{clockTimestamp} = &((\text{hH} * 60 + \text{mM}) * 60 + \text{sS}) * \text{time\_scale} + \\ &\text{nFrames} * (\text{num\_units\_in\_tick} * (1 + \text{nuit\_field\_based\_flag})) + \text{tOffset}, \end{aligned} \tag{D-1}$$

in units of clock ticks of a clock with clock frequency equal to time_scale Hz, relative to some unspecified point in time for which clockTimestamp is equal to 0. Output order and DPB output timing are not affected by the value of clockTimestamp. When two or more frames with pic_struct equal to 0 are consecutive in output order and have equal values of clockTimestamp, the indication is that the frames represent the same content and that the last such frame in output order is the preferred representation.

NOTE 4 – clockTimestamp time indications may aid display on devices with refresh rates other than those well-matched to DPB output times.

clock_timestamp_flag[i **]** equal to 1 indicates that a number of clock timestamp syntax elements are present and follow immediately. clock_timestamp_flag[i] equal to 0 indicates that the associated clock timestamp syntax elements are not present. When NumClockTS is greater than 1 and clock_timestamp_flag[i] is equal to 1 for more than one value of i, the value of clockTimestamp shall be non-decreasing with increasing value of i.

ct_type indicates the scan type (interlaced or progressive) of the source material as specified in Table D-2.

Two fields of a coded frame may have different values of ct_type.

When clockTimestamp is equal for two fields of opposite parity that are consecutive in output order, both with ct_type equal to 0 (progressive) or ct_type equal to 2 (unknown), the two fields are indicated to have come from the same original progressive frame. Two consecutive fields in output order shall have different values of clockTimestamp when the value of ct_type for either field is 1 (interlaced).

Table D-2 – Mapping of ct_type to source picture scan

Value	Original picture scan
0	progressive
1	interlaced
2	unknown
3	reserved

nuit_field_based_flag is used in calculating clockTimestamp, as specified in Equation D-1.

counting_type specifies the method of dropping values of the n_frames as specified in Table D-3.

Table D-3 – Definition of counting_type values

Value	Interpretation
0	no dropping of n_frames count values and no use of time_offset
1	no dropping of n_frames count values
2	dropping of individual zero values of n_frames count
3	dropping of individual MaxFPS-1 values of n_frames count
4	dropping of the two lowest (value 0 and 1) n_frames counts when seconds_value is equal to 0 and minutes_value is not an integer multiple of 10
5	dropping of unspecified individual n_frames count values
6	dropping of unspecified numbers of unspecified n_frames count values
7..31	reserved

full_timestamp_flag equal to 1 specifies that the n_frames syntax element is followed by seconds_value, minutes_value, and hours_value. full_timestamp_flag equal to 0 specifies that the n_frames syntax element is followed by seconds_flag.

discontinuity_flag equal to 0 indicates that the difference between the current value of clockTimestamp and the value of clockTimestamp computed from the previous clock timestamp in output order can be interpreted as the time difference between the times of origin or capture of the associated frames or fields. discontinuity_flag equal to 1 indicates that the difference between the current value of clockTimestamp and the value of clockTimestamp computed from the previous clock timestamp in output order should not be interpreted as the time difference between the times of origin or capture of the associated frames or fields. When discontinuity_flag is equal to 0, the value of clockTimestamp shall be greater than or equal to all values of clockTimestamp present for the preceding picture in DPB output order.

cnt_dropped_flag specifies the skipping of one or more values of n_frames using the counting method specified by counting_type.

n_frames specifies the value of nFrames used to compute clockTimestamp. n_frames shall be less than

$$MaxFPS = Ceil(time\_scale \div num\_units\_in\_tick) \quad (D\text{-}2)$$

NOTE 5 – n_frames is a frame-based counter. For field-specific timing indications, time_offset should be used to indicate a distinct clockTimestamp for each field.

When counting_type is equal to 2 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 1 and the value of n_frames for the previous picture in output order shall not be equal to 0 unless discontinuity_flag is equal to 1.

NOTE 6 – When counting_type is equal to 2, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 12.5 frames per second with time_scale equal to 25 and num_units_in_tick equal to 2 and nuit_field_based_flag equal to 0) can be avoided by occasionally skipping over the value n_frames equal to 0 when counting (e.g., counting n_frames from 0 to 12, then incrementing seconds_value and counting n_frames from 1 to 12, then incrementing seconds_value and counting n_frames from 0 to 12, etc.).

When counting_type is equal to 3 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 0 and the value of n_frames for the previous picture in output order shall not be equal to MaxFPS – 1 unless discontinuity_flag is equal to 1.

NOTE 7 – When counting_type is equal to 3, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 12.5 frames per second with time_scale equal to 25 and num_units_in_tick equal to 2 and nuit_field_based_flag equal to 0) can be avoided by occasionally skipping over the value n_frames equal to MaxFPS when counting (e.g., counting n_frames from 0 to 12, then incrementing seconds_value and counting n_frames from 0 to 11, then incrementing seconds_value and counting n_frames from 0 to 12, etc.).

When counting_type is equal to 4 and cnt_dropped_flag is equal to 1, n_frames shall be equal to 2 and the specified value of sS shall be zero and the specified value of mM shall not be an integer multiple of ten and n_frames for the previous picture in output order shall not be equal to 0 or 1 unless discontinuity_flag is equal to 1.

NOTE 8 – When counting_type is equal to 4, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates (e.g., 30000÷1001 frames per second with time_scale equal to 60000 and num_units_in_tick equal to 1 001 and nuit_field_based_flag equal to 1) can be reduced by occasionally skipping over the value n_frames equal to MaxFPS when counting (e.g., counting n_frames from 0 to 29, then incrementing seconds_value and counting n_frames from 0 to 29, etc., until the seconds_value is zero and minutes_value is not an integer multiple of ten, then counting n_frames from 2 to 29, then incrementing seconds_value and counting n_frames from 0 to 29, etc.). This counting method is well known in industry and is often referred to as "NTSC drop-frame" counting.

When counting_type is equal to 5 or 6 and cnt_dropped_flag is equal to 1, n_frames shall not be equal to 1 plus the value of n_frames for the previous picture in output order modulo MaxFPS unless discontinuity_flag is equal to 1.

NOTE 9 – When counting_type is equal to 5 or 6, the need for increasingly large magnitudes of tOffset in Equation D-1 when using fixed non-integer frame rates can be avoided by occasionally skipping over some values of n_frames when counting. The specific values of n_frames that are skipped are not specified when counting_type is equal to 5 or 6.

seconds_flag equal to 1 specifies that seconds_value and minutes_flag are present when full_timestamp_flag is equal to 0. seconds_flag equal to 0 specifies that seconds_value and minutes_flag are not present.

seconds_value specifies the value of sS used to compute clockTimestamp. The value of seconds_value shall be in the range of 0 to 59, inclusive. When seconds_value is not present, the previous seconds_value in decoding order shall be used as sS to compute clockTimestamp.

minutes_flag equal to 1 specifies that minutes_value and hours_flag are present when full_timestamp_flag is equal to 0 and seconds_flag is equal to 1. minutes_flag equal to 0 specifies that minutes_value and hours_flag are not present.

minutes_value specifies the value of mM used to compute clockTimestamp. The value of minutes_value shall be in the range of 0 to 59, inclusive. When minutes_value is not present, the previous minutes_value in decoding order shall be used as mM to compute clockTimestamp.

hours_flag equal to 1 specifies that hours_value is present when full_timestamp_flag is equal to 0 and seconds_flag is equal to 1 and minutes_flag is equal to 1.

hours_value specifies the value of hH used to compute clockTimestamp. The value of hours_value shall be in the range of 0 to 23, inclusive. When hours_value is not present, the previous hours_value in decoding order shall be used as hH to compute clockTimestamp.

time_offset specifies the value of tOffset used to compute clockTimestamp. The number of bits used to represent time_offset shall be equal to time_offset_length. When time_offset is not present, the value 0 shall be used as tOffset to compute clockTimestamp.

D.2.3 Pan-scan rectangle SEI message semantics

The pan-scan rectangle SEI message syntax elements specify the coordinates of a rectangle relative to the cropping rectangle of the sequence parameter set. Each coordinate of this rectangle is specified in units of one-sixteenth sample spacing relative to the luma sampling grid.

pan_scan_rect_id contains an identifying number that may be used to identify the purpose of the pan-scan rectangle (for example, to identify the rectangle as the area to be shown on a particular display device or as the area that contains a particular actor in the scene). The value of pan_scan_rect_id shall be in the range of 0 to $2^{32} - 1$, inclusive.

Values of pan_scan_rect_id from 0 to 255 and from 512 to 2^{31}-1 may be used as determined by the application. Values of pan_scan_rect_id from 256 to 511 and from 2^{31} to 2^{32}-1 are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of pan_scan_rect_id in the range of 256 to 511 or in the range of 2^{31} to 2^{32} - 1 shall ignore (remove from the bitstream and discard) it.

pan_scan_rect_cancel_flag equal to 1 indicates that the SEI message cancels the persistence of any previous pan-scan rectangle SEI message in output order. pan_scan_rect_cancel_flag equal to 0 indicates that pan-scan rectangle information follows.

pan_scan_cnt_minus1 specifies the number of pan-scan rectangles that are present in the SEI message. pan_scan_cnt_minus1 shall be in the range of 0 to 2, inclusive. pan_scan_cnt_minus1 equal to 0 indicates that a single pan-scan rectangle is present that applies to all fields of the decoded picture. pan_scan_cnt_minus1 shall be equal to 0 when the current picture is a field. pan_scan_cnt_minus1 equal to 1 indicates that two pan-scan rectangles are present, the first of which applies to the first field of the picture in output order and the second of which applies to the second field of the picture in output order. pan_scan_cnt_minus1 equal to 2 indicates that three pan-scan rectangles are present, the first of which applies to the first field of the picture in output order, the second of which applies to the second field of the picture in output order, and the third of which applies to a repetition of the first field as a third field in output order.

pan_scan_rect_left_offset[i], **pan_scan_rect_right_offset[i]**, **pan_scan_rect_top_offset[i]**, and **pan_scan_rect_bottom_offset[i]**, specify, as signed integer quantities in units of one-sixteenth sample spacing relative to the luma sampling grid, the location of the pan-scan rectangle. The values of each of these four syntax elements shall be in the range of -2^{31} to 2^{31} - 1, inclusive.

The pan-scan rectangle is specified, in units of one-sixteenth sample spacing relative to a luma frame sampling grid, as the region with frame horizontal coordinates from 16*CropUnitX * frame_crop_left_offset + pan_scan_rect_left_offset[i] to 16 * (16 * PicWidthInMbs – CropUnitX * frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1 and with vertical coordinates from 16 *CropUnitY * frame_crop_top_offset + pan_scan_rect_top_offset[i] to 16 * (16 * PicHeightInMbs – CropUnitY * frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1, inclusive. The value of 16 * CropUnitX * frame_crop_left_offset + pan_scan_rect_left_offset[i] shall be less than or equal to 16 * (16 * PicWidthInMbs – CropUnitX * frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1; and the value of 16 CropUnitY * frame_crop_top_offset + pan_scan_rect_top_offset[i] shall be less than or equal to 16 * (16 * PicHeightInMbs – CropUnitY * frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1.

When the pan-scan rectangular area includes samples outside of the cropping rectangle, the region outside of the cropping rectangle may be filled with synthesized content (such as black video content or neutral grey video content) for display.

pan_scan_rect_repetition_period specifies the persistence of the pan-scan rectangle SEI message and may specify a picture order count interval within which another pan-scan rectangle SEI message with the same value of pan_scan_rect_id or the end of the coded video sequence shall be present in the bitstream. The value of pan_scan_rect_repetition_period shall be in the range of 0 to 16 384, inclusive. When pan_scan_cnt_minus1 is greater than 0, pan_scan_rect_repetition_period shall not be greater than 1.

pan_scan_rect_repetition_period equal to 0 specifies that the pan-scan rectangle information applies to the current decoded picture only.

pan_scan_rect_repetition_period equal to 1 specifies that the pan-scan rectangle information persists in output order until any of the following conditions are true.

- A new coded video sequence begins.
- A picture in an access unit containing a pan-scan rectangle SEI message with the same value of pan_scan_rect_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

pan_scan_rect_repetition_period equal to 0 or equal to 1 indicates that another pan-scan rectangle SEI message with the same value of pan_scan_rect_id may or may not be present.

pan_scan_rect_repetition_period greater than 1 specifies that the pan-scan rectangle information persists until any of the following conditions are true.

- A new coded video sequence begins.

– A picture in an access unit containing a pan-scan rectangle SEI message with the same value of pan_scan_rect_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period.

pan_scan_rect_repetition_period greater than 1 indicates that another pan-scan rectangle SEI message with the same value of pan_scan_rect_id shall be present for a picture in an access unit that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period; unless the bitstream ends or a new coded video sequence begins without output of such a picture.

D.2.4 Filler payload SEI message semantics

This message contains a series of payloadSize bytes of value 0xFF, which can be discarded.

ff_byte shall be a byte having the value 0xFF.

D.2.5 User data registered by ITU-T Rec. T.35 SEI message semantics

This message contains user data registered as specified by ITU-T Rec. T.35, the contents of which are not specified by this Recommendation | International Standard.

itu_t_t35_country_code shall be a byte having a value specified as a country code by ITU-T Rec. T.35 Annex A.

itu_t_t35_country_code_extension_byte shall be a byte having a value specified as a country code by ITU-T Rec. T.35 Annex B.

itu_t_t35_payload_byte shall be a byte containing data registered as specified by ITU-T Rec. T.35.

The ITU-T T.35 terminal provider code and terminal provider oriented code shall be contained in the first one or more bytes of the itu_t_t35_payload_byte, in the format specified by the Administration that issued the terminal provider code. Any remaining itu_t_t35_payload_byte data shall be data having syntax and semantics as specified by the entity identified by the ITU-T T.35 country code and terminal provider code.

D.2.6 User data unregistered SEI message semantics

This message contains unregistered user data identified by a UUID, the contents of which are not specified by this Recommendation | International Standard.

uuid_iso_iec_11578 shall have a value specified as a UUID according to the procedures of ISO/IEC 11578:1996 Annex A.

user_data_payload_byte shall be a byte containing data having syntax and semantics as specified by the UUID generator.

D.2.7 Recovery point SEI message semantics

The recovery point SEI message assists a decoder in determining when the decoding process will produce acceptable pictures for display after the decoder initiates random access or after the encoder indicates a broken link in the sequence. When the decoding process is started with the access unit in decoding order associated with the recovery point SEI message, all decoded pictures at or subsequent to the recovery point in output order specified in this SEI message are indicated to be correct or approximately correct in content. Decoded pictures produced by random access at or before the picture associated with the recovery point SEI message need not be correct in content until the indicated recovery point, and the operation of the decoding process starting at the picture associated with the recovery point SEI message may contain references to pictures not available in the decoded picture buffer.

In addition, by use of the broken_link_flag, the recovery point SEI message can indicate to the decoder the location of some pictures in the bitstream that can result in serious visual artefacts when displayed, even when the decoding process was begun at the location of a previous IDR access unit in decoding order.

NOTE 1 – The broken_link_flag can be used by encoders to indicate the location of a point after which the decoding process for the decoding of some pictures may cause references to pictures that, though available for use in the decoding process, are not the pictures that were used for reference when the bitstream was originally encoded (e.g., due to a splicing operation performed during the generation of the bitstream).

The recovery point is specified as a count in units of frame_num increments subsequent to the frame_num of the current access unit at the position of the SEI message.

NOTE 2 – When HRD information is present in the bitstream, a buffering period SEI message should be associated with the access unit associated with the recovery point SEI message in order to establish initialisation of the HRD buffer model after a random access.

recovery_frame_cnt specifies the recovery point of output pictures in output order. All decoded pictures in output order are indicated to be correct or approximately correct in content starting at the output order position of the reference picture having the frame_num equal to the frame_num of the VCL NAL units for the current access unit incremented by recovery_frame_cnt in modulo MaxFrameNum arithmetic. recovery_frame_cnt shall be in the range of 0 to MaxFrameNum – 1, inclusive.

exact_match_flag indicates whether decoded pictures at and subsequent to the specified recovery point in output order derived by starting the decoding process at the access unit associated with the recovery point SEI message shall be an exact match to the pictures that would be produced by starting the decoding process at the location of a previous IDR access unit in the NAL unit stream. The value 0 indicates that the match need not be exact and the value 1 indicates that the match shall be exact.

When decoding starts from the location of the recovery point SEI message, all references to not available reference pictures shall be inferred as references to pictures containing only macroblocks coded using Intra macroblock prediction modes and having sample values given by Y samples equal to 128, Cb samples equal to 128, and Cr samples equal to 128 (mid-level grey) for purposes of determining the conformance of the value of exact_match_flag.

NOTE 3 – When performing random access, decoders should infer all references to not available reference pictures as references to pictures containing only intra macroblocks and having sample values given by Y equal to 128, Cb equal to 128, and Cr equal to 128 (mid-level grey), regardless of the value of exact_match_flag.

When exact_match_flag is equal to 0, the quality of the approximation at the recovery point is chosen by the encoding process and is not specified by this Recommendation | International Standard.

broken_link_flag indicates the presence or absence of a broken link in the NAL unit stream at the location of the recovery point SEI message and is assigned further semantics as follows.

- If broken_link_flag is equal to 1, pictures produced by starting the decoding process at the location of a previous IDR access unit may contain undesirable visual artefacts to the extent that decoded pictures at and subsequent to the access unit associated with the recovery point SEI message in decoding order should not be displayed until the specified recovery point in output order.
- Otherwise (broken_link_flag is equal to 0), no indication is given regarding any potential presence of visual artefacts.

Regardless of the value of the broken_link_flag, pictures subsequent to the specified recovery point in output order are specified to be correct or approximately correct in content.

NOTE 4 – When a sub-sequence information SEI message is present in conjunction with a recovery point SEI message in which broken_link_flag is equal to 1 and when sub_seq_layer_num is equal to 0, sub_seq_id should be different from the latest sub_seq_id for sub_seq_layer_num equal to 0 that was decoded prior to the location of the recovery point SEI message. When broken_link_flag is equal to 0, the sub_seq_id in sub-sequence layer 0 should remain unchanged.

changing_slice_group_idc equal to 0 indicates that decoded pictures are correct or approximately correct in content at and subsequent to the recovery point in output order when all macroblocks of the primary coded pictures are decoded within the changing slice group period, i.e., the period between the access unit associated with the recovery point SEI message (inclusive) and the specified recovery point (inclusive) in decoding order. changing_slice_group_idc shall be equal to 0 when num_slice_groups_minus1 is equal to 0 in any primary coded picture within the changing slice group period.

When changing_slice_group_idc is equal to 1 or 2, num_slice_groups_minus1 shall be equal to 1 and the macroblock-to-slice-group map type 3, 4, or 5 shall be applied in each primary coded picture in the changing slice group period.

changing_slice_group_idc equal to 1 indicates that within the changing slice group period no sample values outside the decoded macroblocks covered by slice group 0 are used for inter prediction of any macroblock within slice group 0. In addition, changing_slice_group_idc equal to 1 indicates that when all macroblocks in slice group 0 within the changing slice group period are decoded, decoded pictures will be correct or approximately correct in content at and subsequent to the specified recovery point in output order regardless of whether any macroblock in slice group 1 within the changing slice group period is decoded.

changing_slice_group_idc equal to 2 indicates that within the changing slice group period no sample values outside the decoded macroblocks covered by slice group 1 are used for inter prediction of any macroblock within slice group 1. In addition, changing_slice_group_idc equal to 2 indicates that when all macroblocks in slice group 1 within the changing slice group period are decoded, decoded pictures will be correct or approximately correct in content at and subsequent to the specified recovery point in output order regardless of whether any macroblock in slice group 0 within the changing slice group period is decoded.

changing_slice_group_idc shall be in the range of 0 to 2, inclusive.

D.2.8 Decoded reference picture marking repetition SEI message semantics

The decoded reference picture marking repetition SEI message is used to repeat the decoded reference picture marking syntax structure that was located in the slice header of an earlier picture in the sequence in decoding order.

original_idr_flag shall be equal to 1 when the decoded reference picture marking syntax structure occurred originally in an IDR picture. original_idr_flag shall be equal to 0 when the repeated decoded reference picture marking syntax structure did not occur in an IDR picture originally.

original_frame_num shall be equal to the frame_num of the picture where the repeated decoded reference picture marking syntax structure originally occurred. The picture indicated by original_frame_num is the previous coded picture having the specified value of frame_num. The value of original_frame_num used to refer to a picture having a memory_management_control_operation equal to 5 shall be 0.

original_field_pic_flag shall be equal to the field_pic_flag of the picture where the repeated decoded reference picture marking syntax structure originally occurred.

original_bottom_field_flag shall be equal to the bottom_field_flag of the picture where the repeated decoded reference picture marking syntax structure originally occurred.

dec_ref_pic_marking( ) shall contain a copy of the decoded reference picture marking syntax structure of the picture whose frame_num was original_frame_num. The IdrPicFlag used for specification of the repeated dec_ref_pic_marking( ) syntax structure shall be the IdrPicFlag of the slice header(s) of the picture whose frame_num was original_frame_num (i.e., IdrPicFlag as used in subclause 7.3.3.3 shall be considered equal to original_idr_flag).

D.2.9 Spare picture SEI message semantics

This SEI message indicates that certain slice group map units, called spare slice group map units, in one or more decoded reference pictures resemble the co-located slice group map units in a specified decoded picture called the target picture. A spare slice group map unit may be used to replace a co-located, incorrectly decoded slice group map unit, in the target picture. A decoded picture containing spare slice group map units is called a spare picture.

For all spare pictures identified in a spare picture SEI message, the value of frame_mbs_only_flag shall be equal to the value of frame_mbs_only_flag of the target picture in the same SEI message. The spare pictures in the SEI message are constrained as follows.

- If the target picture is a decoded field, all spare pictures identified in the same SEI message shall be decoded fields.
- Otherwise (the target picture is a decoded frame), all spare pictures identified in the same SEI message shall be decoded frames.

For all spare pictures identified in a spare picture SEI message, the values of pic_width_in_mbs_minus1 and pic_height_in_map_units_minus1 shall be equal to the values of pic_width_in_mbs_minus1 and pic_height_in_map_units_minus1, respectively, of the target picture in the same SEI message. The picture associated (as specified in subclause 7.4.1.2.3) with this message shall appear after the target picture, in decoding order.

target_frame_num indicates the frame_num of the target picture.

spare_field_flag equal to 0 indicates that the target picture and the spare pictures are decoded frames. spare_field_flag equal to 1 indicates that the target picture and the spare pictures are decoded fields.

target_bottom_field_flag equal to 0 indicates that the target picture is a top field. target_bottom_field_flag equal to 1 indicates that the target picture is a bottom field.

A target picture is a decoded reference picture whose corresponding primary coded picture precedes the current picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to target_frame_num, spare_field_flag and target_bottom_field_flag, respectively.

num_spare_pics_minus1 indicates the number of spare pictures for the specified target picture. The number of spare pictures is equal to num_spare_pics_minus1 + 1. The value of num_spare_pics_minus1 shall be in the range of 0 to 15, inclusive.

delta_spare_frame_num[i **]** is used to identify the spare picture that contains the i-th set of spare slice group map units, hereafter called the i-th spare picture, as specified below. The value of delta_spare_frame_num[i] shall be in the range of 0 to MaxFrameNum - 1 - !spare_field_flag, inclusive.

The frame_num of the i-th spare picture, spareFrameNum[i], is derived as follows for all values of i from 0 to num_spare_pics_minus1, inclusive:

```
candidateSpareFrameNum = target_frame_num - !spare_field_flag
for ( i = 0; i <= num_spare_pics_minus1; i++ ) {
    if( candidateSpareFrameNum < 0 )
        candidateSpareFrameNum = MaxFrameNum – 1
    spareFrameNum[ i ] = candidateSpareFrameNum – delta_spare_frame_num[ i ]          (D-3)
    if( spareFrameNum[ i ] < 0 )
        spareFrameNum[ i ] = MaxFrameNum + spareFrameNum[ i ]
    candidateSpareFrameNum = spareFrameNum[ i ] - !spare_field_flag
}
```

spare_bottom_field_flag[i] equal to 0 indicates that the i-th spare picture is a top field. spare_bottom_field_flag[i] equal to 1 indicates that the i-th spare picture is a bottom field.

The 0-th spare picture is a decoded reference picture whose corresponding primary coded picture precedes the target picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to spareFrameNum[0], spare_field_flag and spare_bottom_field_flag[0], respectively. The i-th spare picture is a decoded reference picture whose corresponding primary coded picture precedes the (i - 1)-th spare picture, in decoding order, and in which the values of frame_num, field_pic_flag (when present) and bottom_field_flag (when present) are equal to spareFrameNum[i], spare_field_flag and spare_bottom_field_flag[i], respectively.

spare_area_idc[i] indicates the method used to identify the spare slice group map units in the i-th spare picture. spare_area_idc[i] shall be in the range of 0 to 2, inclusive. spare_area_idc[i] equal to 0 indicates that all slice group map units in the i-th spare picture are spare units. spare_area_idc[i] equal to 1 indicates that the value of the syntax element spare_unit_flag[i][j] is used to identify the spare slice group map units. spare_area_idc[i] equal to 2 indicates that the zero_run_length[i][j] syntax element is used to derive the values of spareUnitFlagInBoxOutOrder[i][j], as described below.

spare_unit_flag[i][j] equal to 0 indicates that the j-th slice group map unit in raster scan order in the i-th spare picture is a spare unit. spare_unit_flag[i][j] equal to 1 indicates that the j-th slice group map unit in raster scan order in the i-th spare picture is not a spare unit.

zero_run_length[i][j] is used to derive the values of spareUnitFlagInBoxOutOrder[i][j] when spare_area_idc[i] is equal to 2. In this case, the spare slice group map units identified in spareUnitFlagInBoxOutOrder[i][j] appear in counter-clockwise box-out order, as specified in subclause 8.2.2.4, for each spare picture. spareUnitFlagInBoxOutOrder[i][j] equal to 0 indicates that the j-th slice group map unit in counter-clockwise box-out order in the i-th spare picture is a spare unit. spareUnitFlagInBoxOutOrder[i][j] equal to 1 indicates that the j-th slice group map unit in counter-clockwise box-out order in the i-th spare picture is not a spare unit.

When spare_area_idc[0] is equal to 2, spareUnitFlagInBoxOutOrder[0][j] is derived as follows:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ 0 ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 0          (D-4)
    spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 1
}
```

When spare_area_idc[i] is equal to 2 and the value of i is greater than 0, spareUnitFlagInBoxOutOrder[i][j] is derived as follows:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
    for( k = 0; k < zero_run_length[ i ][ loop ]; k++ )
        spareUnitFlagInBoxOutOrder[ i ][ j ] = spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]          (D-5)
    spareUnitFlagInBoxOutOrder[ i ][ j ] = !spareUnitFlagInBoxOutOrder[ i - 1 ][ j++ ]
}
```

D.2.10 Scene information SEI message semantics

A scene and a scene transition are herein defined as a set of consecutive pictures in output order.

NOTE 1 – Decoded pictures within one scene generally have similar content. The scene information SEI message is used to label pictures with scene identifiers and to indicate scene changes. The message specifies how the source pictures for the labelled

pictures were created. The decoder may use the information to select an appropriate algorithm to conceal transmission errors. For example, a specific algorithm may be used to conceal transmission errors that occurred in pictures belonging to a gradual scene transition. Furthermore, the scene information SEI message may be used in a manner determined by the application, such as for indexing the scenes of a coded sequence.

A scene information SEI message labels all pictures, in decoding order, from the primary coded picture to which the SEI message is associated (inclusive), as specified in subclause 7.4.1.2.3, to the primary coded picture to which the next scene information SEI message (if present) in decoding order is associated (exclusive) or (otherwise) to the last access unit in the bitstream (inclusive). These pictures are herein referred to as the target pictures.

scene_info_present_flag equal to 0 indicates that the scene or scene transition to which the target pictures belong is unspecified. scene_info_present_flag equal to 1 indicates that the target pictures belong to the same scene or scene transition.

scene_id identifies the scene to which the target pictures belong. When the value of scene_transition_type of the target pictures is less than 4, and the previous picture in output order is marked with a value of scene_transition_type less than 4, and the value of scene_id is the same as the value of scene_id of the previous picture in output order, this indicates that the source scene for the target pictures and the source scene for the previous picture (in output order) are considered by the encoder to have been the same scene. When the value of scene_transition_type of the target pictures is greater than 3, and the previous picture in output order is marked with a value of scene_transition_type less than 4, and the value of scene_id is the same as the value of scene_id of the previous picture in output order, this indicates that one of the source scenes for the target pictures and the source scene for the previous picture (in output order) are considered by the encoder to have been the same scene. When the value of scene_id is not equal to the value of scene_id of the previous picture in output order, this indicates that the target pictures and the previous picture (in output order) are considered by the encoder to have been from different source scenes.

The value of scene_id shall be in the range of 0 to 2^{32}-1, inclusive. Values of scene_id in the range of 0 to 255, inclusive, and in the range of 512 to $2^{31} - 1$, inclusive, may be used as determined by the application. Values of scene_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to $2^{32} - 1$, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of scene_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

scene_transition_type specifies in which type of a scene transition (if any) the target pictures are involved. The valid values of scene_transition_type are specified in Table D-4.

Table D-4 – scene_transition_type values

Value	Description
0	No transition
1	Fade to black
2	Fade from black
3	Unspecified transition from or to constant colour
4	Dissolve
5	Wipe
6	Unspecified mixture of two scenes

When scene_transition_type is greater than 3, the target pictures include contents both from the scene labelled by its scene_id and the next scene, in output order, which is labelled by second_scene_id (see below). The term "the current scene" is used to indicate the scene labelled by scene_id. The term "the next scene" is used to indicate the scene labelled by second_scene_id. It is not required for any following picture, in output order, to be labelled with scene_id equal to second_scene_id of the current SEI message.

Scene transition types are specified as follows.

"No transition" specifies that the target pictures are not involved in a gradual scene transition.

NOTE 2 – When two consecutive pictures in output order have scene_transition_type equal to 0 and different values of scene_id, a scene cut occurred between the two pictures.

"Fade to black" indicates that the target pictures are part of a sequence of pictures, in output order, involved in a fade to black scene transition, i.e., the luma samples of the scene gradually approach zero and the chroma samples of the scene gradually approach 128.

NOTE 3 – When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Fade to black", the later one, in output order, is darker than the previous one.

"Fade from black" indicates that the target pictures are part of a sequence of pictures, in output order, involved in a fade from black scene transition, i.e., the luma samples of the scene gradually diverge from zero and the chroma samples of the scene may gradually diverge from 128.

NOTE 4 – When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Fade from black", the later one in output order is lighter than the previous one.

"Dissolve" indicates that the sample values of each target picture (before encoding) were generated by calculating a sum of co-located weighted sample values of a picture from the current scene and a picture from the next scene. The weight of the current scene gradually decreases from full level to zero level, whereas the weight of the next scene gradually increases from zero level to full level. When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Dissolve", the weight of the current scene for the later one, in output order, is less than the weight of the current scene for the previous one, and the weight of the next scene for the later one, in output order, is greater than the weight of the next scene for the previous one.

"Wipe" indicates that some of the sample values of each target picture (before encoding) were generated by copying co-located sample values of a picture in the current scene and the remaining sample values of each target picture (before encoding) were generated by copying co-located sample values of a picture in the next scene. When two pictures are labelled to belong to the same scene transition and their scene_transition_type is "Wipe", the number of samples copied from the next scene to the later picture in output order is greater than the number of samples copied from the next scene to the previous picture.

second_scene_id identifies the next scene in the gradual scene transition in which the target pictures are involved. The value of second_scene_id shall not be equal to the value of scene_id. The value of second_scene_id shall not be equal to the value of scene_id in the previous picture in output order. When the next picture in output order is marked with a value of scene_transition_type less than 4, and the value of second_scene_id is the same as the value of scene_id of the next picture in output order, this indicates that the encoder considers one of the source scenes for the target pictures and the source scene for the next picture (in output order) to have been the same scene. When the value of second_scene_id is not equal to the value of scene_id or second_scene_id (if present) of the next picture in output order, this indicates that the encoder considers the target pictures and the next picture (in output order) to have been from different source scenes.

When the value of scene_id of a picture is equal to the value of scene_id of the following picture in output order and the value of scene_transition_type in both of these pictures is less than 4, this indicates that the encoder considers the two pictures to have been from the same source scene. When the values of scene_id, scene_transition_type and second_scene_id (if present) of a picture are equal to the values of scene_id, scene_transition_type and second_scene_id (respectively) of the following picture in output order and the value of scene_transition_type is greater than 0, this indicates that the encoder considers the two pictures to have been from the same source gradual scene transition.

The value of second_scene_id shall be in the range of 0 to 2^{32}-1, inclusive. Values of second_scene_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31}-1, inclusive, may be used as determined by the application. Values of second_scene_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of second_scene_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32}-1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.11 Sub-sequence information SEI message semantics

The sub-sequence information SEI message is used to indicate the position of a picture in data dependency hierarchy that consists of sub-sequence layers and sub-sequences.

A sub-sequence layer contains a subset of the coded pictures in a sequence. Sub-sequence layers are numbered with non-negative integers. A layer having a larger layer number is a higher layer than a layer having a smaller layer number. The layers are ordered hierarchically based on their dependency on each other so that any picture in a layer shall not be predicted from any picture on any higher layer.

NOTE 1 – In other words, any picture in layer 0 must not be predicted from any picture in layer 1 or above, pictures in layer 1 may be predicted from layer 0, pictures in layer 2 may be predicted from layers 0 and 1, etc.

NOTE 2 – The subjective quality is expected to increase along with the number of decoded layers.

A sub-sequence is a set of coded pictures within a sub-sequence layer. A picture shall reside in one sub-sequence layer and in one sub-sequence only. Any picture in a sub-sequence shall not be predicted from any picture in another sub-sequence in the same or in a higher sub-sequence layer. A sub-sequence in layer 0 can be decoded independently of any picture that does not belong to the sub-sequence.

The sub-sequence information SEI message concerns the current access unit. The primary coded picture in the access unit is herein referred to as the current picture.

The sub-sequence information SEI message shall not be present unless gaps_in_frame_num_value_allowed_flag in the sequence parameter set referenced by the picture associated with the sub-sequence SEI message is equal to 1.

sub_seq_layer_num specifies the sub-sequence layer number of the current picture. When sub_seq_layer_num is greater than 0, memory management control operations shall not be used in any slice header of the current picture. When the current picture resides in a sub-sequence whose first picture in decoding order is an IDR picture, the value of sub_seq_layer_num shall be equal to 0. For a non-paired reference field, the value of sub_seq_layer_num shall be equal to 0. sub_seq_layer_num shall be in the range of 0 to 255, inclusive.

sub_seq_id identifies the sub-sequence within a layer. When the current picture resides in a sub-sequence whose first picture in decoding order is an IDR picture, the value of sub_seq_id shall be the same as the value of idr_pic_id of the IDR picture. sub_seq_id shall be in the range of 0 to 65535, inclusive.

first_ref_pic_flag equal to 1 specifies that the current picture is the first reference picture of the sub-sequence in decoding order. When the current picture is not the first picture of the sub-sequence in decoding order, the first_ref_pic_flag shall be equal to 0.

leading_non_ref_pic_flag equal to 1 specifies that the current picture is a non-reference picture preceding any reference picture in decoding order within the sub-sequence or that the sub-sequence contains no reference pictures. When the current picture is a reference picture or the current picture is a non-reference picture succeeding at least one reference picture in decoding order within the sub-sequence, the leading_non_ref_pic_flag shall be equal to 0.

last_pic_flag equal to 1 indicates that the current picture is the last picture of the sub-sequence (in decoding order), including all reference and non-reference pictures of the sub-sequence. When the current picture is not the last picture of the sub-sequence (in decoding order), last_pic_flag shall be equal to 0.

The current picture is assigned to a sub-sequence as follows.

- If one or more of the following conditions is true, the current picture is the first picture of a sub-sequence in decoding order.
  - no earlier picture in decoding order is labelled with the same values of sub_seq_id and sub_seq_layer_num as the current picture,
  - the value of leading_non_ref_pic_flag is equal to 1 and the value of leading_non_ref_pic_flag is equal to 0 in the previous picture in decoding order having the same values of sub_seq_id and sub_seq_layer_num as the current picture,
  - the value of first_ref_pic_flag is equal to 1 and the value of leading_non_ref_pic_flag is equal to 0 in the previous picture in decoding order having the same values of sub_seq_id and sub_seq_layer_num as the current picture,
  - the value of last_pic_flag is equal to 1 in the previous picture in decoding order having the same values of sub_seq_id and sub_seq_layer_num as the current picture.
- Otherwise, the current picture belongs to the same sub-sequence as the previous picture in decoding order having the same values of sub_seq_id and sub_seq_layer_num as the current picture.

sub_seq_frame_num_flag equal to 0 specifies that sub_seq_frame_num is not present. sub_seq_frame_num_flag equal to 1 specifies that sub_seq_frame_num is present.

sub_seq_frame_num shall be equal to 0 for the first reference picture of the sub-sequence and for any non-reference picture preceding the first reference picture of the sub-sequence in decoding order. sub_seq_frame_num is further constrained as follows.

- If the current picture is not the second field of a complementary field pair, sub_seq_frame_num shall be incremented by 1, in modulo MaxFrameNum operation, relative to the previous reference picture, in decoding order, that belongs to the sub-sequence.
- Otherwise (the current picture is the second field of a complementary field pair), the value of sub_seq_frame_num shall be the same as the value of sub_seq_frame_num for the first field of the complementary field pair.

sub_seq_frame_num shall be in the range of 0 to MaxFrameNum – 1, inclusive.

When the current picture is an IDR picture, it shall start a new sub-sequence in sub-sequence layer 0. Thus, the sub_seq_layer_num shall be 0, the sub_seq_id shall be different from the previous sub-sequence in sub-sequence layer 0, first_ref_pic_flag shall be 1, and leading_non_ref_pic_flag shall be equal to 0.

When the sub-sequence information SEI message is present for both coded fields of a complementary field pair, the values of sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag and sub_seq_frame_num, when present, shall be identical for both of these pictures.

When the sub-sequence information SEI message is present only for one coded field of a complementary field pair, the values of sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag and sub_seq_frame_num, when present, are also applicable to the other coded field of the complementary field pair.

D.2.12 Sub-sequence layer characteristics SEI message semantics

The sub-sequence layer characteristics SEI message specifies the characteristics of sub-sequence layers.

num_sub_seq_layers_minus1 plus 1 specifies the number of sub-sequence layers in the sequence. num_sub_seq_layers_minus1 shall be in the range of 0 to 255, inclusive.

A pair of average_bit_rate and average_frame_rate characterizes each sub-sequence layer. The first pair of average_bit_rate and average_frame_rate specifies the characteristics of sub-sequence layer 0. When present, the second pair specifies the characteristics of sub-sequence layers 0 and 1 jointly. Each pair in decoding order specifies the characteristics for a range of sub-sequence layers from layer number 0 to the layer number specified by the layer loop counter. The values are in effect from the point they are decoded until an update of the values is decoded.

accurate_statistics_flag equal to 1 indicates that the values of average_bit_rate and average_frame_rate are rounded from statistically correct values. accurate_statistics_flag equal to 0 indicates that the average_bit_rate and the average_frame_rate are estimates and may deviate somewhat from the correct values.

When accurate_statistics_flag is equal to 0, the quality of the approximation used in the computation of the values of average_bit_rate and the average_frame_rate is chosen by the encoding process and is not specified by this Recommendation | International Standard.

average_bit_rate indicates the average bit rate in units of 1000 bits per second. All NAL units in the range of sub-sequence layers specified above are taken into account in the calculation. The average bit rate is derived according to the access unit removal time specified in Annex C of the Recommendation | International Standard. In the following, bTotal is the number of bits in all NAL units succeeding a sub-sequence layer characteristics SEI message (including the bits of the NAL units of the current access unit) and preceding the next access unit (in decoding order) including a sub-sequence layer characteristics SEI message (if present) or the end of the stream (otherwise). t_1 is the removal time (in seconds) of the current access unit, and t_2 is the removal time (in seconds) of the latest access unit (in decoding order) before the next sub-sequence layer characteristics SEI message (if present) or the end of the stream (otherwise).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled as follows.

– If t_1 is not equal to t_2, the following condition shall be true

$$average\_bit\_rate == Round(bTotal \div ((t_2 - t_1) * 1000))) \quad \text{(D-6)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true

$$average\_bit\_rate == 0 \quad \text{(D-7)}$$

average_frame_rate indicates the average frame rate in units of frames/(256 seconds). All NAL units in the range of sub-sequence layers specified above are taken into account in the calculation. In the following, fTotal is the number of frames, complementary field pairs and non-paired fields between the current picture (inclusive) and the next sub-sequence layer characteristics SEI message (if present) or the end of the stream (otherwise). t_1 is the removal time (in seconds) of the current access unit, and t_2 is the removal time (in seconds) of the latest access unit (in decoding order) before the next sub-sequence layer characteristics SEI message (if present) or the end of the stream (otherwise).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled as follows.

– If t_1 is not equal to t_2, the following condition shall be true

$$average\_frame\_rate == Round(fTotal * 256 \div (t_2 - t_1)) \quad \text{(D-8)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true

$$average\_frame\_rate == 0 \quad \text{(D-9)}$$

D.2.13 Sub-sequence characteristics SEI message semantics

The sub-sequence characteristics SEI message indicates the characteristics of a sub-sequence. It also indicates inter prediction dependencies between sub-sequences. This message shall be contained in the first access unit in decoding

order of the sub-sequence to which the sub-sequence characteristics SEI message applies. This sub-sequence is herein called the target sub-sequence.

sub_seq_layer_num identifies the sub-sequence layer number of the target sub-sequence. sub_seq_layer_num shall be in the range of 0 to 255, inclusive.

sub_seq_id identifies the target sub-sequence. sub_seq_id shall be in the range of 0 to 65535, inclusive.

duration_flag equal to 0 indicates that the duration of the target sub-sequence is not specified.

sub_seq_duration specifies the duration of the target sub-sequence in clock ticks of a 90 kHz clock.

average_rate_flag equal to 0 indicates that the average bit rate and the average frame rate of the target sub-sequence are unspecified.

accurate_statistics_flag indicates how reliable the values of average_bit_rate and average_frame_rate are. accurate_statistics_flag equal to 1, indicates that the average_bit_rate and the average_frame_rate are rounded from statistically correct values. accurate_statistics_flag equal to 0 indicates that the average_bit_rate and the average_frame_rate are estimates and may deviate from the statistically correct values.

average_bit_rate indicates the average bit rate in (1000 bits)/second of the target sub-sequence. All NAL units of the target sub-sequence are taken into account in the calculation. The average bit rate is derived according to the access unit removal time specified in subclause C.1.2. In the following, nB is the number of bits in all NAL units in the sub-sequence. t_1 is the removal time (in seconds) of the first access unit of the sub-sequence (in decoding order), and t_2 is the removal time (in seconds) of the last access unit of the sub-sequence (in decoding order).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled as follows.

– If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_bit\_rate} == \text{Round}(nB \div ((t_2 - t_1) * 1000)) \quad \text{(D-10)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_bit\_rate} == 0 \quad \text{(D-11)}$$

average_frame_rate indicates the average frame rate in units of frames/(256 seconds) of the target sub-sequence. All NAL units of the target sub-sequence are taken into account in the calculation. The average frame rate is derived according to the access unit removal time specified in subclause C.1.2. In the following, fC is the number of frames, complementary field pairs and non-paired fields in the sub-sequence. t_1 is the removal time (in seconds) of the first access unit of the sub-sequence (in decoding order), and t_2 is the removal time (in seconds) of the last access unit of the sub-sequence (in decoding order).

When accurate_statistics_flag is equal to 1, the following conditions shall be fulfilled as follows.

– If t_1 is not equal to t_2, the following condition shall be true

$$\text{average\_frame\_rate} == \text{Round}(fC * 256 \div (t_2 - t_1)) \quad \text{(D-12)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true

$$\text{average\_frame\_rate} == 0 \quad \text{(D-13)}$$

num_referenced_subseqs specifies the number of sub-sequences that contain pictures that are used as reference pictures for inter prediction in the pictures of the target sub-sequence. num_referenced_subseqs shall be in the range of 0 to 255, inclusive.

ref_sub_seq_layer_num, ref_sub_seq_id, and **ref_sub_seq_direction** identify the sub-sequence that contains pictures that are used as reference pictures for inter prediction in the pictures of the target sub-sequence. Depending on ref_sub_seq_direction, the following applies.

– If ref_sub_seq_direction is equal to 0, a set of candidate sub-sequences consists of the sub-sequences whose sub_seq_id is equal to ref_sub_seq_id, which reside in the sub-sequence layer having sub_seq_layer_num equal to ref_sub_seq_layer_num, and whose first picture in decoding order precedes the first picture of the target sub-sequence in decoding order.

– Otherwise (ref_sub_seq_direction is equal to 1), a set of candidate sub-sequences consists of the sub-sequences whose sub_seq_id is equal to ref_sub_seq_id, which reside in the sub-sequence layer having sub_seq_layer_num equal to ref_sub_seq_layer_num, and whose first picture in decoding order succeeds the first picture of the target sub-sequence in decoding order.

The sub-sequence used as a reference for the target sub-sequence is the sub-sequence among the set of candidate sub-sequences whose first picture is the closest to the first picture of the target sub-sequence in decoding order.

D.2.14 Full-frame freeze SEI message semantics

The full-frame freeze SEI message indicates that the current picture and any subsequent pictures in output order that meet specified conditions should not affect the content of the display. No more than one full-frame freeze SEI message shall be present in any access unit.

full_frame_freeze_repetition_period specifies the persistence of the full-frame freeze SEI message and may specify a picture order count interval within which another full-frame freeze SEI message or a full-frame freeze release SEI or the end of the coded video sequence shall be present in the bitstream. The value of full_frame_freeze_repetition_period shall be in the range of 0 to 16 384, inclusive.

full_frame_freeze_repetition_period equal to 0 specifies that the full-frame freeze SEI message applies to the current decoded picture only.

full_frame_freeze_repetition_period equal to 1 specifies that the full-frame freeze SEI message persists in output order until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a full-frame freeze SEI message or a full-frame freeze release SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

full_frame_freeze_repetition_period greater than 1 specifies that the full-frame freeze SEI message persists until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a full-frame freeze SEI message or a full-frame freeze release SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period.

full_frame_freeze_repetition_period greater than 1 indicates that another full-frame freeze SEI message or a full-frame freeze release SEI message shall be present for a picture in an access unit that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period; unless the bitstream ends or a new coded video sequence begins without output of such a picture.

D.2.15 Full-frame freeze release SEI message semantics

The full-frame freeze release SEI message cancels the effect of any full-frame freeze SEI message sent with pictures that precede the current picture in output order. The full-frame freeze release SEI message indicates that the current picture and subsequent pictures in output order should affect the contents of the display.

No more than one full-frame freeze release SEI message shall be present in any access unit. A full-frame freeze release SEI message shall not be present in an access unit containing a full-frame freeze SEI message. When a full-frame freeze SEI message is present in an access unit containing a field of a complementary field pair in which the values of PicOrderCnt(CurrPic) for the two fields of the complementary field pair are equal to each other, a full-frame freeze release SEI message shall not be present in either of the two access units.

D.2.16 Full-frame snapshot SEI message semantics

The full-frame snapshot SEI message indicates that the current frame is labelled for use as determined by the application as a still-image snapshot of the video content.

snapshot_id specifies a snapshot identification number. snapshot_id shall be in the range of 0 to 2^{32} - 1, inclusive.

Values of snapshot_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31}-1, inclusive, may be used as determined by the application. Values of snapshot_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of snapshot_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32}-1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.17 Progressive refinement segment start SEI message semantics

The progressive refinement segment start SEI message specifies the beginning of a set of consecutive coded pictures that is labelled as the current picture followed by a sequence of one or more pictures of refinement of the quality of the current picture, rather than as a representation of a continually moving scene.

The tagged set of consecutive coded pictures shall continue until one of the following conditions is true. When a condition below becomes true, the next slice to be decoded does not belong to the tagged set of consecutive coded pictures.

1. The next slice to be decoded belongs to an IDR picture.
2. num_refinement_steps_minus1 is greater than 0 and the frame_num of the next slice to be decoded is (currFrameNum + num_refinement_steps_minus1 + 1) % MaxFrameNum, where currFrameNum is the value of frame_num of the picture in the access unit containing the SEI message.
3. num_refinement_steps_minus1 is 0 and a progressive refinement segment end SEI message with the same progressive_refinement_id as the one in this SEI message is decoded.

The decoding order of picture within the tagged set of consecutive pictures should be the same as their output order. **progressive_refinement_id** specifies an identification number for the progressive refinement operation. progressive_refinement_id shall be in the range of 0 to 2^{32} - 1, inclusive.

Values of progressive_refinement_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31} - 1, inclusive, may be used as determined by the application. Values of progressive_refinement_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32}-1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of progressive_refinement_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

num_refinement_steps_minus1 specifies the number of reference frames in the tagged set of consecutive coded pictures as follows.

– If num_refinement_steps_minus1 is equal to 0, the number of reference frames in the tagged set of consecutive coded pictures is unknown.

– Otherwise, the number of reference frames in the tagged set of consecutive coded pictures is equal to num_refinement_steps_minus1 + 1.

num_refinement_steps_minus1 shall be in the range of 0 to MaxFrameNum - 1, inclusive.

D.2.18 Progressive refinement segment end SEI message semantics

The progressive refinement segment end SEI message specifies the end of a set of consecutive coded pictures that has been labelled by use of a progressive refinement segment start SEI message as an initial picture followed by a sequence of one or more pictures of the refinement of the quality of the initial picture, and ending with the current picture.

progressive_refinement_id specifies an identification number for the progressive refinement operation. progressive_refinement_id shall be in the range of 0 to 2^{32} - 1, inclusive.

The progressive refinement segment end SEI message specifies the end of any progressive refinement segment previously started using a progressive refinement segment start SEI message with the same value of progressive_refinement_id.

Values of progressive_refinement_id in the range of 0 to 255, inclusive, and in the range of 512 to 2^{31} - 1, inclusive, may be used as determined by the application. Values of progressive_refinement_id in the range of 256 to 511, inclusive, and in the range of 2^{31} to 2^{32} - 1, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of progressive_refinement_id in the range of 256 to 511, inclusive, or in the range of 2^{31} to 2^{32} - 1, inclusive, shall ignore (remove from the bitstream and discard) it.

D.2.19 Motion-constrained slice group set SEI message semantics

This SEI message indicates that inter prediction over slice group boundaries is constrained as specified below. When present, the message shall only appear where it is associated, as specified in subclause 7.4.1.2.3, with an IDR access unit.

The target picture set for this SEI message contains all consecutive primary coded pictures in decoding order starting with the associated primary coded IDR picture (inclusive) and ending with the following primary coded IDR picture (exclusive) or with the very last primary coded picture in the bitstream (inclusive) in decoding order when there is no following primary coded IDR picture. The slice group set is a collection of one or more slice groups, identified by the slice_group_id[i] syntax element. When separate_colour_plane_flag is equal to 1, the term “primary coded pictures”

represents the parts of the corresponding primary coded pictures that correspond to the NAL units having the same colour_plane_id.

This SEI message indicates that, for each picture in the target picture set, the inter prediction process is constrained as follows: No sample value outside the slice group set, and no sample value at a fractional sample position that is derived using one or more sample values outside the slice group set is used to inter predict any sample within the slice group set.

num_slice_groups_in_set_minus1 + 1 specifies the number of slice groups in the slice group set. The allowed range of num_slice_groups_in_set_minus1 is 0 to num_slice_groups_minus1, inclusive. The allowed range of num_slice_groups_minus1 is specified in Annex A.

slice_group_id[i **]** identifies the slice group(s) contained within the slice group set. The allowed range is from 0 to num_slice_groups_in_set_minus1, inclusive. The size of the slice_group_id[i] syntax element is Ceil(Log2(num_slice_groups_minus1 + 1)) bits.

exact_sample_value_match_flag equal to 0 indicates that, within the target picture set, when the macroblocks that do not belong to the slice group set are not decoded, the value of each sample in the slice group set need not be exactly the same as the value of the same sample when all the macroblocks are decoded. exact_sample_value_match_flag equal to 1 indicates that, within the target picture set, when the macroblocks that do not belong to the slice group set are not decoded, the value of each sample in the slice group set shall be exactly the same as the value of the same sample when all the macroblocks in the target picture set are decoded.

NOTE 1 – When disable_deblocking_filter_idc is equal to 2 in all slices in the target picture set, exact_sample_value_match_flag should be 1.

pan_scan_rect_flag equal to 0 specifies that pan_scan_rect_id is not present. pan_scan_rect_flag equal to 1 specifies that pan_scan_rect_id is present.

pan_scan_rect_id indicates that the specified slice group set covers at least the pan-scan rectangle identified by pan_scan_rect_id within the target picture set.

NOTE 2 – Multiple motion_constrained_slice_group_set SEI messages may be associated with the same IDR picture. Consequently, more than one slice group set may be active within a target picture set.

NOTE 3 – The size, shape, and location of the slice groups in the slice group set may change within the target picture set.

D.2.20 Film grain characteristics SEI message semantics

This SEI message provides the decoder with a parameterised model for film grain synthesis. For example, an encoder may use the film grain characteristics SEI message to characterise film grain that was present in the original source video material and was removed by pre-processing filtering techniques. Synthesis of simulated film grain on the decoded images for the display process is optional and does not affect the decoding process specified in this Recommendation | International Standard. If synthesis of simulated film grain on the decoded images for the display process is performed, there is no requirement that the method by which the synthesis is performed be the same as the parameterised model for the film grain as provided in the film grain characteristics SEI message.

NOTE 1 – The display process is not specified in this Recommendation | International Standard.

film_grain_characteristics_cancel_flag equal to 1 indicates that the SEI message cancels the persistence of any previous film grain characteristics SEI message in output order. film_grain_characteristics_cancel_flag equal to 0 indicates that film grain modelling information follows.

model_id identifies the film grain simulation model as specified in Table D-5. The value of model_id shall be in the range of 0 to 1, inclusive.

Table D-5 – model_id values

Value	Description
0	frequency filtering
1	auto-regression
2	reserved
3	reserved

separate_colour_description_present_flag equal to 1 indicates that a distinct colour space description for the film grain characteristics specified in the SEI message is present in the film grain characteristics SEI message syntax. separate_colour_description_present_flag equal to 0 indicates that the colour description for the film grain characteristics specified in the SEI message is the same as for the coded video sequence as specified in subclause E.2.1.

NOTE 2 – When separate_colour_description_present_flag is equal to 1, the colour space specified for the film grain characteristics specified in the SEI message may differ from the colour space specified for the coded video as specified in subclause E.2.1.

film_grain_bit_depth_luma_minus8 plus 8 specifies the bit depth used for the luma component of the film grain characteristics specified in the SEI message. When film_grain_bit_depth_luma_minus8 is not present in the film grain characteristics SEI message, the value of film_grain_bit_depth_luma_minus8 shall be inferred to be equal to bit_depth_luma_minus8.

The value of filmGrainBitDepth[0] is derived as

$$filmGrainBitDepth[\ 0\ ] = film\_grain\_bit\_depth\_luma\_minus8 + 8 \tag{D-14}$$

film_grain_bit_depth_chroma_minus8 plus 8 specifies the bit depth used for the Cb and Cr components of the film grain characteristics specified in the SEI message. When film_grain_bit_depth_chroma_minus8 is not present in the film grain characteristics SEI message, the value of film_grain_bit_depth_chroma_minus8 shall be inferred to be equal to bit_depth_chroma_minus8.

The value of filmGrainBitDepth[c] for c = 1 and 2 is derived as

$$filmGrainBitDepth[\ c\ ] = film\_grain\_bit\_depth\_chroma\_minus8 + 8 \quad \text{with } c = 1, 2 \tag{D-15}$$

film_grain_full_range_flag has the same semantics as specified in subclause E.2.1 for the video_full_range_flag syntax element, except as follows.

– film_grain_full_range_flag specifies the colour space of the film grain characteristics specified in the SEI message, rather than the colour space used for the coded video sequence.

– When film_grain_full_range_flag is not present in the film grain characteristics SEI message, the value of film_grain_full_range_flag shall be inferred to be equal to video_full_range_flag.

film_grain_colour_primaries has the same semantics as specified in subclause E.2.1 for the colour_primaries syntax element, except as follows.

– film_grain_colour_primaries specifies the colour space of the film grain characteristics specified in the SEI message, rather than the colour space used for the coded video sequence.

– When film_grain_colour_primaries is not present in the film grain characteristics SEI message, the value of film_grain_colour_primaries shall be inferred to be equal to colour_primaries.

film_grain_transfer_characteristics has the same semantics as specified in subclause E.2.1 for the transfer_characteristics syntax element, except as follows.

– film_grain_transfer_characteristics specifies the colour space of the film grain characteristics specified in the SEI message, rather than the colour space used for the coded video sequence.

– When film_grain_transfer_characteristics is not present in the film grain characteristics SEI message, the value of film_grain_transfer_characteristics shall be inferred to be equal to transfer_characteristics.

film_grain_matrix_coefficients has the same semantics as specified in subclause E.2.1 for the matrix_coefficients syntax element, except as follows.

– film_grain_matrix_coefficients specifies the colour space of the film grain characteristics specified in the SEI message, rather than the colour space used for the coded video sequence.

– When film_grain_matrix_coefficients is not present in the film grain characteristics SEI message, the value of film_grain_matrix_coefficients shall be inferred to be equal to matrix_coefficients.

– The values allowed for film_grain_matrix_coefficients are not constrained by the value of chroma_format_idc.

The chroma_format_idc of the film grain characteristics specified in the film grain characteristics SEI message shall be inferred to be equal to 3 (4:4:4).

NOTE 3 – Because the use of a specific method is not required for performing film grain generation function used by the display process, a decoder may, if desired, down-convert the model information for chroma in order to simulate film grain for other chroma formats (4:2:0 or 4:2:2) rather than up-converting the decoded video (using a method not specified by this Recommendation | International Standard) before performing film grain generation.

blending_mode_id identifies the blending mode used to blend the simulated film grain with the decoded images as specified in Table D-6. blending_mode_id shall be in the range of 0 to 1, inclusive.

Table D-6 – blending_mode_id values

Value	Description
0	additive
1	multiplicative
2	reserved
3	reserved

Depending on blending_mode_id, the blending mode is specified as follows

– If blending_mode_id is equal to 0 the blending mode is additive as specified by

$$I_{grain}[x, y, c] = Clip3(0, (1 << filmGrainBitDepth[c]) - 1, I_{decoded}[x, y, c] + G[x, y, c]) \quad \text{(D-16)}$$

– Otherwise (blending_mode_id is equal to 1), the blending mode is multiplicative as specified by

$$I_{grain}[x, y, c] = Clip3(0, (1 << filmGrainBitDepth[c]) - 1, I_{decoded}[x, y, c] * (1 + G[x, y, c])) \quad \text{(D-17)}$$

where $I_{decoded}[x, y, c]$ represents the sample value at coordinates x, y of the colour component c of the decoded image $I_{decoded}$, $G[x, y, c]$ is the simulated film grain value at the same position and colour component, and filmGrainBitDepth[c] is the number of bits used for each sample in a fixed-length unsigned binary representation of the array $I_{grain}[x, y, c]$.

log2_scale_factor specifies a scale factor used in the film grain characterization equations.

comp_model_present_flag[c] equal to 0 indicates that film grain is not modelled on the c-th colour component, where c equal to 0 refers to the luma component, c equal to 1 refers to the Cb component, and c equal to 2 refers to the Cr component. comp_model_present_flag[c] equal to 1 indicates that syntax elements specifying modelling of film grain on colour component c are present in the SEI message.

num_intensity_intervals_minus1[c] plus 1 specifies the number of intensity intervals for which a specific set of model values has been estimated.

NOTE 4 – The intensity intervals may overlap in order to simulate multi-generational film grain.

num_model_values_minus1[c] plus 1 specifies the number of model values present for each intensity interval in which the film grain has been modelled. The value of num_model_values_minus1[c] shall be in the range of 0 to 5, inclusive.

intensity_interval_lower_bound[c][i] specifies the lower bound of the interval i of intensity levels for which the set of model values applies.

intensity_interval_upper_bound[c][i] specifies the upper bound of the interval i of intensity levels for which the set of model values applies.

Depending on model_id, the selection of the sets of model values is specified as follows.

– If model_id is equal to 0, the average value of each block b of 16x16 samples in $I_{decoded}$, referred as b_{avg}, is used to select the sets of model values with index s[j] that apply to all the samples in the block:

```
for( i = 0, j = 0; i <= num_intensity_intervals_minus1; i++ )
   if( bavg >= intensity_interval_lower_bound[ c ][ i ] && bavg <= intensity_interval_upper_bound[ c ][ i ] ) {
     s[ j ] = i                                                                                              (D-18)
     j++
   }
```

– Otherwise (model_id is equal to 1), the sets of model values used to generate the film grain are selected for each sample value in $I_{decoded}$ as follows:

```
for( i = 0, j = 0; i <= num_intensity_intervals_minus1; i++ )
   if( I_decoded[ x, y, c ] >= intensity_interval_lower_bound[ c ][ i ] &&
       I_decoded[ x, y, c ] <= intensity_interval_upper_bound[ c ][ i ] ) {
    s[ j ] = i
    j++
   }
```
(D-19)

Samples that do not fall into any of the defined intervals are not modified by the grain generation function. Samples that fall into more than one interval will originate multi-generation grain. Multi-generation grain results from adding the grain computed independently for each intensity interval.

comp_model_value[c][i][j] represents each one of the model values present for the colour component c and the intensity interval i. The set of model values has different meaning depending on the value of model_id. The value of comp_model_value[c][i][j] shall be constrained as follows, and may be additionally constrained as specified elsewhere in this subclause.

– If model_id is equal to 0, comp_model_value[c][i][j] shall be in the range of 0 to $2^{filmGrainBitDepth[\,c\,]} - 1$, inclusive.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][j] shall be in the range of $-2^{(\,filmGrainBitDepth[\,c\,]-1\,)}$ to $2^{(\,filmGrainBitDepth[\,c\,]-1\,)} - 1$, inclusive.

Depending on model_id, the synthesis of the film grain is modelled as follows.

– If model_id is equal to 0, a frequency filtering model enables simulating the original film grain for c = 0..2, x = 0..$PicWidthInSamples_L$, and y = 0..$PicHeightInSamples_L$ as specified by:

```
G[ x, y, c ] = ( comp_model_value[ c ][ s ][ 0 ] * Q[ x, y, c ] + comp_model_value[ c ][ s ][ 5 ] *
                                        G[ x, y, c-1 ] ) >> log2_scale_factor
```
(D-20)

where Q[c] is a two-dimensional random process generated by filtering 16x16 blocks gaussRv with random-value elements $gaussRv_{ij}$ generated with a normalized Gaussian distribution (independent and identically distributed Gaussian random variable samples with zero mean and unity variance) and where the value of an element G[x, y, c-1] used in the right-hand side of the equation is inferred to be equal to 0 when c-1 is less than 0.

NOTE 5 – A normalized Gaussian random value can be generated from two independent, uniformly distributed random values over the interval from 0 to 1 (and not equal to 0), denoted as uRv_0 and uRv_1, using the Box-Muller transformation specified by

$$gaussRv_{ij} = \sqrt{-2 * Ln(uRv_0)} * Cos(2 * \pi * uRv_1) \quad \text{(D-21)}$$

where Ln(x) is the natural logarithm of x (the base-e logarithm, where e is natural logarithm base constant 2.718 281 828...), Cos(x) is the trigonometric cosine function operating on an argument x in units of radians, and π is Archimedes' constant 3.141 592 653....

The band-pass filtering of blocks gaussRv may be performed in the discrete cosine transform (DCT) domain as follows:

```
for( y = 0; y < 16; y++ )
    for( x = 0: x < 16; x++ )
        if( ( x < comp_model_value[ c ][ s ][ 3 ]  &&  y < comp_model_value[ c ][ s ][ 4 ] ) ||
          x > comp_model_value[ c ][ s ][ 1 ] || y > comp_model_value[ c ][ s ][ 2 ] )
            gaussRv[ x, y ] = 0
filteredRv = IDCT16x16( gaussRv )
```
(D-22)

where IDCT16x16(z) refers to a unitary inverse discrete cosine transformation (IDCT) operating on a 16x16 matrix argument z as specified by

$$IDCT16x16(z) = r * z * r^T \quad \text{(D-23)}$$

where the superscript T indicates a matrix transposition and r is the 16x16 matrix with elements r_{ij} specified by

$$r_{ij} = \frac{((i == 0)\,?\,1 : \sqrt{2})}{4} \mathrm{Cos}\left(\frac{i*(2*j+1)*\pi}{32}\right) \tag{D-24}$$

where Cos(x) is the trigonometric cosine function operating on an argument x in units of radians and π is Archimedes' constant 3.141 592 653.

Q[c] is formed by the frequency-filtered blocks filteredRv.

NOTE 6 – Coded model values are based on blocks of 16x16, but a decoder implementation may use other block sizes. For example, decoders implementing the IDCT on 8x8 blocks, should down-convert by a factor of two the set of coded model values comp_model_value[c][s][i] for i equal to 1..4.

NOTE 7 – To reduce the degree of visible blocks that can result from mosaicing the frequency-filtered blocks filteredRv, decoders may apply a low-pass filter to the boundaries between frequency-filtered blocks.

– Otherwise (model_id is equal to 1), an auto-regression model enables simulating the original film grain for c = 0..2, x = 0..$PicWidthInSamples_L$, and y = 0..$PicHeightInSamples_L$ as specified by

```
G[ x, y, c ] = ( comp_model_value[ c ][ s ][ 0 ] * n[ x, y, c ] +
    comp_model_value[ c ][ s ][ 1 ] * ( G[ x-1, y, c ] + ( ( comp_model_value[ c ][ s ][ 4 ] * G[ x, y-1, c ] ) >>
        log2_scale_factor ) ) +
    comp_model_value[ c ][ s ][ 3 ] * ( ( ( comp_model_value[ c ][ s ][ 4 ] * G[ x-1, y-1, c ] ) >>
        log2_scale_factor ) + G[ x+1, y-1, c ] ) +
    comp_model_value[ c ][ s ][ 5 ] * ( G[ x-2, y, c ] +
        ( ( comp_model_value[ c ][ s ][ 4 ] * comp_model_value[ c ][ s ][ 4 ] * G[ x, y-2, c ] ) >>
            ( 2 * log2_scale_factor ) ) ) +
    comp_model_value[ c ][ s ][ 2 ] * G[ x, y, c-1 ] ) >> log2_scale_factor                    (D-25)
```

where n[x, y, c] is a random value with normalized Gaussian distribution (independent and identically distributed Gaussian random variable samples with zero mean and unity variance for each value of x, y, and c) and where the value of an element G[x, y, c] used in the right-hand side of the equation is inferred to be equal to 0 when any of the following conditions are true:

– x is less than 0,

– y is less than 0,

– x is greater than or equal to $PicWidthInSamples_L$,

– c is less than 0.

comp_model_value[c][i][0] provides the first model value for the model as specified by model_id. comp_model_value[c][i][0] corresponds to the standard deviation of the Gaussian noise term in the generation functions specified in Equations D-20 through D-23.

comp_model_value[c][i][1] provides the second model value for the model as specified by model_id. comp_model_value[c][i][1] shall be greater than or equal to 0 and less than 16.

When not present in the film grain characteristics SEI message, comp_model_value[c][i][1] shall be inferred as follows.

– If model_id is equal to 0, comp_model_value[c][i][1] shall be inferred to be equal to 8.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][1] shall be inferred to be equal to 0.

comp_model_value[c][i][1] is interpreted as follows.

– If model_id is equal to 0, comp_model_value[c][i][1] indicates the horizontal high cut frequency to be used to filter the DCT of a block of 16x16 random values.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][1] indicates the first order spatial correlation for neighbouring samples (x-1, y) and (x, y-1).

comp_model_value[c][i][2] provides the third model value for the model as specified by model_id. comp_model_value[c][i][2] shall be greater than or equal to 0 and less than 16.

When not present in the film grain characteristics SEI message, comp_model_value[c][i][2] shall be inferred as follows.

– If model_id is equal to 0, comp_model_value[c][i][2] shall be inferred to be equal to comp_model_value[c][i][1].

– Otherwise (model_id is equal to 0), comp_model_value[c][i][2] shall be inferred to be equal to 0.

comp_model_value[c][i][2] is interpreted as follows.

– If model_id is equal to 0, comp_model_value[c][i][2] indicates the vertical high cut frequency to be used to filter the DCT of a block of 16x16 random values.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][2] indicates the colour correlation between consecutive colour components.

comp_model_value[c][i][3] provides the fourth model value for the model as specified by model_id. comp_model_value[c][i][3] shall be greater than or equal to 0 and less than or equal to comp_model_value[c][i][1].

When not present in the film grain characteristics SEI message, comp_model_value[c][i][3] shall be inferred to be equal to 0.

comp_model_value[c][i][3] is interpreted as follows.

– If model_id is equal to 0, comp_model_value[c][i][3] indicates the horizontal low cut frequency to be used to filter the DCT of a block of 16x16 random values.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][3] indicates the first order spatial correlation for neighbouring samples (x-1, y-1) and (x+1, y-1).

comp_model_value[c][i][4] provides the fifth model value for the model as specified by model_id. comp_model_value[c][i][4] shall be greater than or equal to 0 and less than or equal to comp_model_value[c][i][2].

When not present in the film grain characteristics SEI message, comp_model_value[c][i][4] shall be inferred to be equal to model_id.

comp_model_value[c][i][4] is interpreted as follows.

– If model_id is equal to 0, comp_model_value[c][i][4] indicates the vertical low cut frequency to be used to filter the DCT of a block of 16x16 random values.

– Otherwise (model id is equal to 1), comp model value[c][i][4] indicates the aspect ratio of the modelled grain.

comp_model_value[c][i][5] provides the sixth model value for the model as specified by model_id. When not present in the film grain characteristics SEI message, comp_model_value[c][i][5] shall be inferred to be equal to 0.

comp_model_value[c][i][5] is interpreted as follows.

– If model_id is equal to 0, comp_model_value[c][i][5] indicates the colour correlation between consecutive colour components.

– Otherwise (model_id is equal to 1), comp_model_value[c][i][5] indicates the second order spatial correlation for neighbouring samples (x, y-2) and (x-2, y).

film_grain_characteristics_repetition_period specifies the persistence of the film grain characteristics SEI message and may specify a picture order count interval within which another film grain characteristics SEI message or the end of the coded video sequence shall be present in the bitstream. The value of film_grain_characteristics_repetition_period shall be in the range 0 to 16 384, inclusive.

film_grain_characteristics_repetition_period equal to 0 specifies that the film grain characteristics SEI message applies to the current decoded picture only.

film_grain_characteristics_repetition_period equal to 1 specifies that the film grain characteristics SEI message persists in output order until any of the following conditions are true.

– A new coded video sequence begins, or

– A picture in an access unit containing a film grain characteristics SEI message that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

film_grain_characteristics_repetition_period greater than 1 specifies that the film grain characteristics SEI message persists until any of the following conditions are true.

– A new coded video sequence begins, or

– A picture in an access unit containing a film grain characteristics SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + film_grain_characteristics_repetition_period.

film_grain_characteristics_repetition_period greater than 1 indicates that another film grain characteristics SEI message shall be present for a picture in an access unit that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + film_grain_characteristics_repetition_period; unless the bitstream ends or a new coded video sequence begins without output of such a picture.

D.2.21 Deblocking filter display preference SEI message semantics

This SEI message provides the decoder with an indication of whether the display of the cropped result of the deblocking filter process specified in subclause 8.7 or of the cropped result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 is preferred by the encoder for the display of each decoded picture that is output.

NOTE 1 – The display process is not specified in this Recommendation | International Standard. The means by which an encoder determines what to indicate as its preference expressed in a deblocking filter display preference SEI message is also not specified in this Recommendation | International Standard, and the expression of an expressed preference in a deblocking filter display preference SEI message does not impose any requirement on the display process.

deblocking_display_preference_cancel_flag equal to 1 indicates that the SEI message cancels the persistence of any previous deblocking filter display preference SEI message in output order. deblocking_display_preference_cancel_flag equal to 0 indicates that a display_prior_to_deblocking_preferred_flag and deblocking_display_preference_repetition_period follow.

NOTE 2 – In the absence of the deblocking filter display preference SEI message, or after the receipt of a deblocking filter display preference SEI message in which deblocking_display_preference_cancel_flag is equal to 1, the decoder should infer that the display of the cropped result of the deblocking filter process specified in subclause 8.7 is preferred over the display of the cropped result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 for the display of each decoded picture that is output.

display_prior_to_deblocking_preferred_flag equal to 1 indicates that the encoder preference is for the display process (which is not specified in this Recommendation | International Standard) to display the cropped result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 rather than the cropped result of the deblocking filter process specified in subclause 8.7 for each picture that is cropped and output as specified in Annex C. display_prior_to_deblocking_preferred_flag equal to 0 indicates that the encoder preference is for the display process (which is not specified in this Recommendation | International Standard) to display the cropped result of the deblocking filter process specified in subclause 8.7 rather than the cropped result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 for each picture that is cropped and output as specified in Annex C.

NOTE 3 – The presence or absence of the deblocking filter display preference SEI message and the value of display_prior_to_deblocking_preferred_flag does not affect the requirements of the decoding process specified in this Recommendation | International Standard. Rather, it only provides an indication of when, in addition to fulfilling the requirements of this Recommendation | International Standard for the decoding process, enhanced visual quality might be obtained by performing the display process (which is not specified in this Recommendation | International Standard) in an alternative fashion. Encoders that use the deblocking filter display preference SEI message should be designed with an awareness that unless the encoder restricts its use of the DPB capacity specified in Annex A for the profile and level in use, some decoders may not have sufficient memory capacity for the storage of the result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 in addition to the storage of the result of the deblocking filter process specified in subclause 8.7 when reordering and delaying pictures for display, and such decoders would therefore not be able to benefit from the preference indication. By restricting its use of the DPB capacity, an encoder can be able to use at least half of the DPB capacity specified in Annex A while allowing the decoder to use the remaining capacity for storage of unfiltered pictures that have been indicated as preferable for display until the output time arrives for those pictures.

dec_frame_buffering_constraint_flag equal to 1 indicates that the use of the frame buffering capacity of the HRD decoded picture buffer (DPB) as specified by max_dec_frame_buffering has been constrained such that the coded video sequence will not require a decoded picture buffer with more than Max(1, max_dec_frame_buffering) frame buffers to enable the output of the decoded filtered or unfiltered pictures, as indicated by the deblocking filter display preference SEI messages, at the output times specified by the dpb_output_delay of the picture timing SEI messages. dec_frame_buffering_constraint_flag equal to 0 indicates that the use of the frame buffering capacity in the HRD may or may not be constrained in the manner that would be indicated by dec_frame_buffering_constraint_flag equal to 1.

For purposes of determining the constraint imposed when dec_frame_buffering_constraint_flag is equal to 1, the quantity of frame buffering capacity used at any given point in time by each frame buffer of the DPB that contains a picture shall be derived as follows:

- If both of the following criteria are satisfied for the frame buffer, the frame buffer is considered to use two frame buffers of capacity for its storage.
- The frame buffer contains a frame or one or more fields that is marked as "used for reference", and
- The frame buffer contains a picture for which both of the following criteria are fulfilled.
  - The HRD output time of the picture is greater than the given point in time, and

– It has been indicated in a deblocking filter display preference SEI message that the encoder preference for the picture is for the display process to display the cropped result of the picture construction process prior to the deblocking filter process specified in subclause 8.5.13 rather than the cropped result of the deblocking filter process specified in subclause 8.7, and

– Otherwise, the frame buffer is considered to use one frame buffer of DPB capacity for its storage.

When dec_frame_buffering_constraint_flag is equal to 1, the frame buffering capacity used by all of the frame buffers in the DPB that contain pictures, as derived in this manner, shall not be greater than Max(1, max_dec_frame_buffering) during the operation of the HRD for the coded video sequence.

The value of dec_frame_buffering_constraint_flag shall be the same in all deblocking filter display preference SEI messages of the coded video sequence.

deblocking_display_preference_repetition_period specifies the persistence of the film grain characteristics SEI message and may specify a picture order count interval within which another film grain characteristics SEI message or the end of the coded video sequence shall be present in the bitstream. The value of deblocking_display_preference_repetition_period shall be in the range 0 to 16 384, inclusive.

deblocking_display_preference_repetition_period equal to 0 specifies that the deblocking filter display preference SEI message applies to the current decoded picture only.

deblocking_display_preference_repetition_period equal to 1 specifies that the deblocking filter display preference SEI message persists in output order until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a deblocking filter display preference SEI message that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

deblocking_display_preference_repetition_period greater than 1 specifies that the deblocking filter display preference SEI message persists until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a deblocking filter display preference SEI message is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + deblocking_display_preference_repetition_period.

deblocking_display_preference_repetition_period greater than 1 indicates that another deblocking filter display preference SEI message shall be present for a picture in an access unit that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + deblocking_display_preference_repetition_period; unless the bitstream ends or a new coded video sequence begins without output of such a picture.

D.2.22 Stereo video information SEI message semantics

This SEI message provides the decoder with an indication that the entire coded video sequence consists of pairs of pictures forming stereo-view content.

The stereo video information SEI message shall not be present in any access unit of a coded video sequence unless a stereo video information SEI message is present in the first access unit of the coded video sequence.

field_views_flag equal to 1 indicates that all pictures in the current coded video sequence are fields and all fields of a particular parity are considered a left view and all fields of the opposite parity are considered a right view for stereo-view content. field_views_flag equal to 0 indicates that all pictures in the current coded video sequence are frames and alternating frames in output order represent a view of a stereo view. The value of field_views_flag shall be the same in all stereo video information SEI messages within a coded video sequence.

When the stereo video information SEI message is present and field_views_flag is equal to 1, the left view and right view of a stereo video pair shall be coded as a complementary field pair, the display time of the first field of the field pair in output order should be delayed to coincide with the display time of the second field of the field pair in output order, and the spatial locations of the samples in each individual field should be interpreted for display purposes as representing complete pictures as shown in Figure 6-1 rather than as spatially-distinct fields within a frame as shown in Figure 6-2.

NOTE – The display process is not specified in this Recommendation | International Standard.

top_field_is_left_view_flag equal to 1 indicates that the top fields in the coded video sequence represent a left view and the bottom fields in the coded video sequence represent a right view. top_field_is_left_view_flag equal to 0 indicates that the bottom fields in the coded video sequence represent a left view and the top fields in the coded video sequence represent a right view. When present, the value of top_field_is_left_view_flag shall be the same in all stereo video information SEI messages within a coded video sequence.

current_frame_is_left_view_flag equal to 1 indicates that the current picture is the left view of a stereo-view pair. current_frame_is_left_view_flag equal to 0 indicates that the current picture is the right view of a stereo-view pair.

next_frame_is_second_view_flag equal to 1 indicates that the current picture and the next picture in output order form a stereo-view pair, and the display time of the current picture should be delayed to coincide with the display time of the next picture in output order. next_frame_is_second_view_flag equal to 0 indicates that the current picture and the previous picture in output order form a stereo-view pair, and the display time of the current picture should not be delayed for purposes of stereo-view pairing.

left_view_self_contained_flag equal to 1 indicates that no inter prediction operations within the decoding process for the left-view pictures of the coded video sequence refer to reference pictures that are right-view pictures. left_view_self_contained_flag equal to 0 indicates that some inter prediction operations within the decoding process for the left-view pictures of the coded video sequence may or may not refer to reference pictures that are right-view pictures. Within a coded video sequence, the value of left_view_self_contained_flag in all stereo video information SEI messages shall be the same.

right_view_self_contained_flag equal to 1 indicates that no inter prediction operations within the decoding process for the right-view pictures of the coded video sequence refer to reference pictures that are left-view pictures. right_view_self_contained_flag equal to 0 indicates that some inter prediction operations within the decoding process for the right-view pictures of the coded video sequence may or may not refer to reference pictures that are left-view pictures. Within a coded video sequence, the value of right_view_self_contained_flag in all stereo video information SEI messages shall be the same.

D.2.23 Post-filter hint SEI message semantics

This SEI message provides the coefficients of a post-filter or correlation information for the design of a post-filter for potential use in post-processing of the output decoded pictures to obtain improved displayed quality.

filter_hint_size_y specifies the vertical size of the filter coefficient or correlation array. The value of filter_hint_size_y shall be in the range of 1 to 15, inclusive.

filter_hint_size_x specifies the horizontal size of the filter coefficient or correlation array. The value of filter_hint_size_x shall be in the range of 1 to 15, inclusive.

filter_hint_type identifies the type of the transmitted filter hints as specified in Table D-7. The value of filter_hint_type shall be in the range of 0 to 2, inclusive. Decoders that conform to this Recommendation | International Standard shall ignore post-filter hint SEI messages having filter_hint_type equal to the reserved value 3.

Table D-7 – filter_hint_type values

Value	Description
0	coefficients of a 2D FIR filter
1	coefficients of two 1D FIR filters
2	cross-correlation matrix
3	Reserved

filter_hint[colour_component **][** cy **][** cx **]** specifies a filter coefficient or an element of a cross-correlation matrix between original and decoded signal with 16-bit precision. colour_component specifies the related colour component. cy represents a counter in vertical direction, cx represents a counter in horizontal direction. Depending on filter_hint_type, the following applies.

– If filter_hint_type is equal to 0, the coefficients of a 2-dimensional FIR filter with the size of filter_hint_size_y * filter_hint_size_x are transmitted.

– Otherwise, if filter_hint_type is equal to 1, the filter coefficients of two 1-dimensional FIR filters are transmitted. In this case, filter_hint_size_y shall be equal to 2. The index cy = 0 specifies the filter coefficients of the horizontal filter and cy = 1 specifies the filter coefficients of the vertical filter. In the filtering process, the horizontal filter shall be applied first and the result shall be filtered by the vertical filter.

– Otherwise (filter_hint_type is equal to 2), the transmitted hints specify a cross-correlation matrix between the original signal s and the decoded signal s'.

NOTE 1 - The cross-correlation matrix for a related colour component with the size of filter_hint_size_y * filter_hint_size_x is defined as follows:

$$\text{filter\_hint}(\text{cy},\text{cx}) = \frac{1}{h \cdot w} \sum_{m=0}^{h-1} \sum_{n=0}^{w-1} s(m,n) \cdot s'(m + \text{cy} - \mathit{offset\_y}, n + \text{cx} - \mathit{offset\_x}) \quad \text{(D-26)}$$

where *s* denotes the original frame, *s'* denotes the decoded frame, *h* denotes the vertical height of the related colour component, *w* denotes the horizontal width of the related colour component, *offset_y* is equal to (filter_hint_size_y >> 1), *offset_x* is equal to (filter_hint_size_x >> 1), 0 <= cy < filter_hint_size_y and 0 <= cx < filter_hint_size_x.

NOTE 2 - A decoder can derive a Wiener post-filter from the cross-correlation matrix of original and decoded signal and the auto-correlation matrix of the decoded signal.

additional_extension_flag equal to 0 indicates that no additional data follows within the post-filter hint SEI message. The value of additional_extension_flag shall be equal to 0. The value of 1 for additional_extension_flag is reserved for future use by ITU-T | ISO/IEC. Decoders that conform to this Recommendation | International Standard shall ignore all data that follows the value of 1 for additional_extension_flag in a post-filter hint SEI message.

D.2.24 Tone mapping information SEI message semantics

This SEI message provides information to enable remapping of the colour samples of the output decoded pictures for customization to particular display environments. The remapping process maps coded sample values in the RGB colour space (specified in Annex E) to target sample values. All mappings are expressed in the RGB colour space and should be applied to each RGB component separately.

tone_map_id contains an identifying number that may be used to identify the purpose of the tone mapping model. The value of tone_map_id shall be in the range of 0 to $2^{32} - 1$, inclusive.

Values of tone_map_id from 0 to 255 and from 512 to $2^{31} - 1$ may be used as determined by the application. Values of tone_map_id from 256 to 511 and from 2^{31} to $2^{32} - 1$ are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore (remove from the bitstream and discard) all tone mapping information SEI messages containing a value of tone_map_id in the range of 256 to 511 or in the range of 2^{31} to $2^{32} - 1$, and bitstreams shall not contain such values.

NOTE 1 – The tone_map_id can be used to support tone mapping operations that are suitable for different display scenarios. For example, different values of tone_map_id may correspond to different display bit depths.

tone_map_cancel_flag equal to 1 indicates that the tone mapping information SEI message cancels the persistence of any previous tone mapping information SEI message in output order. tone_map_cancel_flag equal to 0 indicates that tone mapping information follows.

tone_map_repetition_period specifies the persistence of the tone mapping information SEI message and may specify a picture order count interval within which another tone mapping information SEI message with the same value of tone_map _id or the end of the coded video sequence shall be present in the bitstream. The value of tone_map_repetition_period shall be in the range of 0 to 16 384, inclusive.

tone_map_repetition_period equal to 0 specifies that the tone map information applies to the current decoded picture only.

tone_map_repetition_period equal to 1 specifies that the tone map information persists in output order until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a tone mapping information SEI message with the same value of tone_map_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic).

tone_map_repetition_period equal to 0 or equal to 1 indicates that another tone mapping information SEI message with the same value of tone_map_id may or may not be present.

tone_map_repetition_period greater than 1 specifies that the tone map information persists until any of the following conditions are true.

– A new coded video sequence begins.

– A picture in an access unit containing a tone mapping information SEI message with the same value of tone_map_id is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + tone_map_repetition_period.

tone_map_repetition_period greater than 1 indicates that another tone mapping information SEI message with the same value of tone_map_id shall be present for a picture in an access unit that is output having PicOrderCnt( ) greater than PicOrderCnt(CurrPic) and less than or equal to PicOrderCnt(CurrPic) + tone_map_repetition_period; unless the bitstream ends or a new coded video sequence begins without output of such a picture.

coded_data_bit_depth specifies the $BitDepth_Y$ of the luma component of the coded video sequence. It is used to identify the tone mapping information SEI message that is intended for use with the coded video sequence. If tone mapping information SEI messages are present that have coded_data_bit_depth that is not equal to $BitDepth_Y$, these refer to the hypothetical result of a transcoding operation performed to convert the coded video to the $BitDepth_Y$ corresponding to the value of coded_data_bit_depth.

The value of coded_data_bit_depth shall be in the range of 8 to 14, inclusive. Values of coded_data_bit_depth from 0 to 7 and from 15 to 255 are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore (remove from the bitstream and discard) all tone mapping SEI messages that contain a coded_data_bit_depth in the range of 0 to 7 or in the range of 15 to 255, and bitstreams shall not contain such values.

target_bit_depth specifies the bit depth of the output of the dynamic range mapping function (or tone mapping function) described by the tone mapping information SEI message. The tone mapping function specified with a particular target_bit_depth is suggested to be reasonable for all display bit depths that are equal to or less than the target_bit_depth.

The value of target_bit_depth shall be in the range of 1 to 16, inclusive. Values of target_bit_depth equal to 0 and in the range of 17 to 255 are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore (remove from the bitstream and discard) all tone mapping SEI messages that contain a value of target_bit_depth equal to 0 or in the range of 17 to 255, and bitstreams shall not contain such values.

model_id specifies the model utilized for mapping the coded data into the target_bit_depth range. Values greater than 3 are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore (remove from the bitstream and discard) all tone mapping SEI messages that contain a value of model_id greater than 3, and bitstreams shall not contain such values.

NOTE 2 – A model_id of 0 corresponds to a linear mapping with clipping; a model_id of 1 corresponds to a sigmoidal mapping; a model_id of 2 corresponds to a user-defined table mapping, and a model_id of 3 corresponds to a piece-wise linear mapping.

min_value specifies the RGB sample value in the coded data that maps to the minimum value in the signalled target_bit_depth. It is used in combination with the max_value parameter. All values in the coded data that are less than or equal to min_value are mapped to this minimum value in the target_bit_depth representation.

max_value specifies the RGB sample value in the coded data that maps to the maximum value in the signalled target_bit_depth. It is used in combination with the min_value parameter. All values in the coded data that are larger than or equal to max_value are mapped to this maximum value in the target_bit_depth representation.

sigmoid_midpoint specifies the RGB sample value of the coded data that is mapped to the center point of the target_bit_depth representation. It is used in combination with the sigmoid_width parameter.

sigmoid_width specifies the distance between two coded data values that approximately correspond to the 5% and 95% values of the target_bit_depth representation, respectively. It is used in combination with the sigmoid_midpoint parameter and is interpreted according to the following function:

$$f(i) = Round\left(\frac{2^{target\_bit\_depth} - 1}{1 + \exp\left(\frac{-6 * (i - sigmoid\_midpoint)}{sigmoid\_width} \right)} \right) \quad for\ i = 0, \cdots, (2^{coded\_bit\_depth} - 1) \qquad \text{(D-27)}$$

where $f($ i $)$ denotes the function that maps an RGB sample value i from the coded data to a resulting RGB sample value in the target_bit_depth representation.

start_of_coded_interval[i **]** specifies the beginning point of an interval in the coded data such that all RGB sample values that are greater than or equal to start_of_coded_interval[i] and less than start_of_coded_interval[i + 1] are mapped to i in the target bit depth representation. The value of start_of_coded_interval[$2^{target\_bit\_depth}$] is equal to $2^{coded\_bit\_depth}$. The number of bits used for the representation of the start_of_coded_interval is ((coded_data_bit_depth + 7) >> 3) << 3.

num_pivots specifies the number of pivot points in the piece-wise linear mapping function without counting the two default end points, (0, 0) and ($2^{coded\_data\_bit\_depth} - 1$, $2^{target\_bit\_depth} - 1$) .

coded_pivot_value[i] specifies the value in the coded_bit_depth corresponding to the i-th pivot point. The number of bits used for the representation of the coded_pivot_value is ((coded_data_bit_depth + 7) >> 3) << 3.

target_pivot_value[i] specifies the value in the reference target_bit_depth corresponding to the i-th pivot point. The number of bits used for the representation of the target_pivot_value is ((target_bit_depth + 7) >> 3) << 3.

D.2.25 Reserved SEI message semantics

This message consists of data reserved for future backward-compatible use by ITU-T | ISO/IEC. Encoders conforming to this Recommendation | International Standard shall not send reserved SEI messages until and unless the use of such messages has been specified by ITU-T | ISO/IEC. Decoders conforming to this Recommendation | International Standard that encounter reserved SEI messages shall discard their content without effect on the decoding process, except as specified in future Recommendations | International Standards specified by ITU-T | ISO/IEC.

reserved_sei_message_payload_byte is a byte reserved for future use by ITU-T | ISO/IEC.

Annex E
Video usability information

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

This Annex specifies syntax and semantics of the VUI parameters of the sequence parameter sets.

VUI parameters are not required for constructing the luma or chroma samples by the decoding process. Conforming decoders are not required to process this information for output order conformance to this Recommendation | International Standard (see Annex C for the specification of conformance). Some VUI parameters are required to check bitstream conformance and for output timing decoder conformance.

In Annex E, specification for presence of VUI parameters is also satisfied when those parameters (or some subset of them) are conveyed to decoders (or to the HRD) by other means not specified by this Recommendation | International Standard. When present in the bitstream, VUI parameters shall follow the syntax and semantics specified in subclauses 7.3.2.1 and 7.4.2.1 and this annex. When the content of VUI parameters is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the VUI parameters is not required to use the same syntax specified in this annex. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

E.1 VUI syntax

E.1.1 VUI parameters syntax

vui_parameters( ) {	C	Descriptor
aspect_ratio_info_present_flag	0	u(1)
if(aspect_ratio_info_present_flag) {		
aspect_ratio_idc	0	u(8)
if(aspect_ratio_idc == Extended_SAR) {		
sar_width	0	u(16)
sar_height	0	u(16)
}		
}		
overscan_info_present_flag	0	u(1)
if(overscan_info_present_flag)		
overscan_appropriate_flag	0	u(1)
video_signal_type_present_flag	0	u(1)
if(video_signal_type_present_flag) {		
video_format	0	u(3)
video_full_range_flag	0	u(1)
colour_description_present_flag	0	u(1)
if(colour_description_present_flag) {		
colour_primaries	0	u(8)
transfer_characteristics	0	u(8)
matrix_coefficients	0	u(8)
}		
}		
chroma_loc_info_present_flag	0	u(1)
if(chroma_loc_info_present_flag) {		
chroma_sample_loc_type_top_field	0	ue(v)
chroma_sample_loc_type_bottom_field	0	ue(v)
}		

timing_info_present_flag	0	u(1)
if(timing_info_present_flag) {		
num_units_in_tick	0	u(32)
time_scale	0	u(32)
fixed_frame_rate_flag	0	u(1)
}		
nal_hrd_parameters_present_flag	0	u(1)
if(nal_hrd_parameters_present_flag)		
hrd_parameters( )		
vcl_hrd_parameters_present_flag	0	u(1)
if(vcl_hrd_parameters_present_flag)		
hrd_parameters( )		
if(nal_hrd_parameters_present_flag || vcl_hrd_parameters_present_flag)		
low_delay_hrd_flag	0	u(1)
pic_struct_present_flag	0	u(1)
bitstream_restriction_flag	0	u(1)
if(bitstream_restriction_flag) {		
motion_vectors_over_pic_boundaries_flag	0	u(1)
max_bytes_per_pic_denom	0	ue(v)
max_bits_per_mb_denom	0	ue(v)
log2_max_mv_length_horizontal	0	ue(v)
log2_max_mv_length_vertical	0	ue(v)
num_reorder_frames	0	ue(v)
max_dec_frame_buffering	0	ue(v)
}		
}		

E.1.2 HRD parameters syntax

hrd_parameters( ) {	C	Descriptor
cpb_cnt_minus1	0	ue(v)
bit_rate_scale	0	u(4)
cpb_size_scale	0	u(4)
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
bit_rate_value_minus1[SchedSelIdx]	0	ue(v)
cpb_size_value_minus1[SchedSelIdx]	0	ue(v)
cbr_flag[SchedSelIdx]	0	u(1)
}		
initial_cpb_removal_delay_length_minus1	0	u(5)
cpb_removal_delay_length_minus1	0	u(5)
dpb_output_delay_length_minus1	0	u(5)
time_offset_length	0	u(5)
}		

E.2 VUI semantics

E.2.1 VUI parameters semantics

aspect_ratio_info_present_flag equal to 1 specifies that aspect_ratio_idc is present. aspect_ratio_info_present_flag equal to 0 specifies that aspect_ratio_idc is not present.

aspect_ratio_idc specifies the value of the sample aspect ratio of the luma samples. Table E-1 shows the meaning of the code. When aspect_ratio_idc indicates Extended_SAR, the sample aspect ratio is represented by sar_width and sar_height. When the aspect_ratio_idc syntax element is not present, aspect_ratio_idc value shall be inferred to be equal to 0.

Table E-1 – Meaning of sample aspect ratio indicator

aspect_ratio_idc	Sample aspect ratio	(informative) Examples of use
0	Unspecified	
1	1:1 ("square")	1280x720 16:9 frame without horizontal overscan 1920x1080 16:9 frame without horizontal overscan (cropped from 1920x1088) 640x480 4:3 frame without horizontal overscan
2	12:11	720x576 4:3 frame with horizontal overscan 352x288 4:3 frame without horizontal overscan
3	10:11	720x480 4:3 frame with horizontal overscan 352x240 4:3 frame without horizontal overscan
4	16:11	720x576 16:9 frame with horizontal overscan 528x576 4:3 frame without horizontal overscan
5	40:33	720x480 16:9 frame with horizontal overscan 528x480 4:3 frame without horizontal overscan
6	24:11	352x576 4:3 frame without horizontal overscan 480x576 16:9 frame with horizontal overscan
7	20:11	352x480 4:3 frame without horizontal overscan 480x480 16:9 frame with horizontal overscan
8	32:11	352x576 16:9 frame without horizontal overscan
9	80:33	352x480 16:9 frame without horizontal overscan
10	18:11	480x576 4:3 frame with horizontal overscan
11	15:11	480x480 4:3 frame with horizontal overscan
12	64:33	528x576 16:9 frame without horizontal overscan
13	160:99	528x480 16:9 frame without horizontal overscan
14	4:3	1440x1080 16:9 frame without horizontal overscan
15	3:2	1280x1080 16:9 frame without horizontal overscan
16	2:1	960x1080 16:9 frame without horizontal overscan
17..254	Reserved	
255	Extended_SAR	

sar_width indicates the horizontal size of the sample aspect ratio (in arbitrary units).

sar_height indicates the vertical size of the sample aspect ratio (in the same arbitrary units as sar_width).

sar_width and sar_height shall be relatively prime or equal to 0. When aspect_ratio_idc is equal to 0 or sar_width is equal to 0 or sar_height is equal to 0, the sample aspect ratio shall be considered unspecified by this Recommendation | International Standard.

overscan_info_present_flag equal to 1 specifies that the overscan_appropriate_flag is present. When overscan_info_present_flag is equal to 0 or is not present, the preferred display method for the video signal is unspecified.

overscan_appropriate_flag equal to 1 indicates that the cropped decoded pictures output are suitable for display using overscan. overscan_appropriate_flag equal to 0 indicates that the cropped decoded pictures output contain visually important information in the entire region out to the edges of the cropping rectangle of the picture, such that the cropped decoded pictures output should not be displayed using overscan. Instead, they should be displayed using either an exact match between the display area and the cropping rectangle, or using underscan.

NOTE 1 – For example, overscan_appropriate_flag equal to 1 might be used for entertainment television programming, or for a live view of people in a videoconference, and overscan_appropriate_flag equal to 0 might be used for computer screen capture or security camera content.

video_signal_type_present_flag equal to 1 specifies that video_format, video_full_range_flag and colour_description_present_flag are present. video_signal_type_present_flag equal to 0, specify that video_format, video_full_range_flag and colour_description_present_flag are not present.

video_format indicates the representation of the pictures as specified in Table E-2, before being coded in accordance with this Recommendation | International Standard. When the video_format syntax element is not present, video_format value shall be inferred to be equal to 5.

Table E-2 – Meaning of video_format

video_format	Meaning
0	Component
1	PAL
2	NTSC
3	SECAM
4	MAC
5	Unspecified video format
6	Reserved
7	Reserved

video_full_range_flag indicates the black level and range of the luma and chroma signals as derived from E'_Y, E'_{PB}, and E'_{PR} or E'_R, E'_G, and E'_B analogue component signals.

When the video_full_range_flag syntax element is not present, the value of video_full_range_flag shall be inferred to be equal to 0.

colour_description_present_flag equal to 1 specifies that colour_primaries, transfer_characteristics and matrix_coefficients are present. colour_description_present_flag equal to 0 specifies that colour_primaries, transfer_characteristics and matrix_coefficients are not present.

colour_primaries indicates the chromaticity coordinates of the source primaries as specified in Table E-3 in terms of the CIE 1931 definition of x and y as specified by ISO/CIE 10527.

When the colour_primaries syntax element is not present, the value of colour_primaries shall be inferred to be equal to 2 (the chromaticity is unspecified or is determined by the application).

Table E-3 – Colour primaries

Value	Primaries			Informative Remark
0	Reserved			For future use by ITU-T | ISO/IEC
1	primary green blue red white D65	x 0.300 0.150 0.640 0.3127	y 0.600 0.060 0.330 0.3290	ITU-R Rec. BT.709-5 ITU-R Rec. BT.1361 conventional colour gamut system and extended colour gamut system IEC 61966-2-4 Society of Motion Picture and Television Engineers RP 177 (1993) Annex B
2	Unspecified			Image characteristics are unknown or are determined by the application.
3	Reserved			For future use by ITU-T | ISO/IEC
4	primary green blue red white C	x 0.21 0.14 0.67 0.310	y 0.71 0.08 0.33 0.316	ITU-R Rec. BT.470-6 System M (historical) United States National Television System Committee 1953 Recommendation for transmission standards for colour television United States Federal Communications Commission Title 47 Code of Federal Regulations (2003) 73.682 (a) (20)
5	primary green blue red white D65	x 0.29 0.15 0.64 0.3127	y 0.60 0.06 0.33 0.3290	ITU-R Rec. BT.470-6 System B, G (historical) ITU-R Rec. BT.601-6 625 ITU-R Rec. BT.1358 625 ITU-R Rec. BT.1700 625 PAL and 625 SECAM
6	primary green blue red white D65	x 0.310 0.155 0.630 0.3127	y 0.595 0.070 0.340 0.3290	ITU-R Rec. BT.601-6 525 ITU-R Rec. BT.1358 525 ITU-R Rec. BT.1700 NTSC Society of Motion Picture and Television Engineers 170M (2004) (functionally the same as the value 7)
7	primary green blue red white D65	x 0.310 0.155 0.630 0.3127	y 0.595 0.070 0.340 0.3290	Society of Motion Picture and Television Engineers 240M (1999) (functionally the same as the value 6)
8	primary green blue red white C	x 0.243 0.145 0.681 0.310	y 0.692 (Wratten 58) 0.049 (Wratten 47) 0.319 (Wratten 25) 0.316	Generic film (colour filters using Illuminant C)
9-255	Reserved			For future use by ITU-T | ISO/IEC

transfer_characteristics indicates the opto-electronic transfer characteristic of the source picture as specified in Table E-4 as a function of a linear optical intensity input L_c with an analogue range of 0 to 1.

When the transfer_characteristics syntax element is not present, the value of transfer_characteristics shall be inferred to be equal to 2 (the transfer characteristics are unspecified or are determined by the application).

Table E-4 – Transfer characteristics

Value	Transfer Characteristic		Informative Remark
0	Reserved		For future use by ITU-T | ISO/IEC
1	$V = 1.099 * L_c^{0.45} - 0.099$ $V = 4.500 * L_c$	for $1 >= L_c >= 0.018$ for $0.018 > L_c >= 0$	ITU-R Rec. BT.709-5 ITU-R Rec. BT.1361 conventional colour gamut system (functionally the same as the value 6)
2	Unspecified		Image characteristics are unknown or are determined by the application.
3	Reserved		For future use by ITU-T | ISO/IEC
4	Assumed display gamma 2.2		ITU-R Rec. BT.470-6 System M (historical) United States National Television System Committee 1953 Recommendation for transmission standards for colour television United States Federal Communications Commission Title 47 Code of Federal Regulations (2003) 73.682 (a) (20) ITU-R Rec. BT.1700 (2007 revision) 625 PAL and 625 SECAM
5	Assumed display gamma 2.8		ITU-R Rec. BT.470-6 System B, G (historical)
6	$V = 1.099 * L_c^{0.45} - 0.099$ $V = 4.500 * L_c$	for $1 >= L_c >= 0.018$ for $0.018 > L_c >= 0$	ITU-R Rec. BT.601-6 525 or 625 ITU-R Rec. BT.1358 525 or 625 ITU-R Rec. BT.1700 NTSC Society of Motion Picture and Television Engineers 170M (2004) (functionally the same as the value 1)
7	$V = 1.1115 * L_c^{0.45} - 0.1115$ $V = 4.0 * L_c$	for $1 >= L_c >= 0.0228$ for $0.0228 > L_c >= 0$	Society of Motion Picture and Television Engineers 240M (1999)
8	$V = L_c$	for $1 > L_c >= 0$	Linear transfer characteristics
9	$V = 1.0 - \text{Log10}(L_c) \div 2$ $V = 0.0$	for $1 >= L_c >= 0.01$ for $0.01 > L_c >= 0$	Logarithmic transfer characteristic (100:1 range)
10	$V = 1.0 - \text{Log10}(L_c) \div 2.5$ $V = 0.0$	for $1 >= L_c >= 0.0031622777$ for $0.0031622777 > L_c >= 0$	Logarithmic transfer characteristic (316.22777:1 range)
11	$V = 1.099 * L_c^{0.45} - 0.099$ $V = 4.500 * L_c$ $V = -1.099 * (-L_c)^{0.45} + 0.099$	for $L_c >= 0.018$ for $0.018 > L_c > -0.018$ for $-0.018 >= L_c$	IEC 61966-2-4
12	$V = 1.099 * L_c^{0.45} - 0.099$ $V = 4.500 * L_c$ $V = -(1.099 * (-4 * L_c)^{0.45} - 0.099) \div 4$	for $1.33 > L_c >= 0.018$ for $0.018 > L_c >= -0.0045$ for $-0.0045 > L_c >= -0.25$	ITU-R Rec. BT.1361 extended colour gamut system
13..255	Reserved		For future use by ITU-T | ISO/IEC

matrix_coefficients describes the matrix coefficients used in deriving luma and chroma signals from the green, blue, and red primaries, as specified in Table E-5.

matrix_coefficients shall not be equal to 0 unless both of the following conditions are true

– BitDepth_C is equal to BitDepth_Y

– chroma_format_idc is equal to 3 (4:4:4)

The specification of the use of matrix_coefficients equal to 0 under all other conditions is reserved for future use by ITU-T | ISO/IEC.

matrix_coefficients shall not be equal to 8 unless one of the following conditions are true

– $BitDepth_C$ is equal to $BitDepth_Y$

– $BitDepth_C$ is equal to $BitDepth_Y + 1$ and chroma_format_idc is equal to 3 (4:4:4)

The specification of the use of matrix_coefficients equal to 8 under all other conditions is reserved for future use by ITU-T | ISO/IEC.

When the matrix_coefficients syntax element is not present, the value of matrix_coefficients shall be inferred to be equal to 2 (unspecified).

The interpretation of matrix_coefficients is specified as follows.

– If transfer_characteristics is not equal to 11 or 12, E'_R, E'_G, and E'_B are analog with values in the range of 0 to 1.

– Otherwise (transfer_characteristics is equal to 11 (IEC 61966-2-4) or 12 (ITU-R BT.1361 extended colour gamut system)), E'_R, E'_G and E'_B are analog with a larger range not specified in this Recommendation.

– Nominal white is specified as having E'_R equal to 1, E'_G equal to 1, and E'_B equal to 1.

– Nominal black is specified as having E'_R equal to 0, E'_G equal to 0, and E'_B equal to 0.

– If video_full_range_flag is equal to 0, the following equations apply.

  – If matrix_coefficients is equal to 1, 4, 5, 6, or 7, the following equations apply.

$$Y = Clip1_Y(Round((1 << (BitDepth_Y - 8)) * (219 * E'_Y + 16))) \quad \text{(E-1)}$$

$$Cb = Clip1_C(Round((1 << (BitDepth_C - 8)) * (224 * E'_{PB} + 128))) \quad \text{(E-2)}$$

$$Cr = Clip1_C(Round((1 << (BitDepth_C - 8)) * (224 * E'_{PR} + 128))) \quad \text{(E-3)}$$

  – Otherwise, if matrix_coefficients is equal to 0 or 8, the following equations apply.

$$R = Clip1_Y((1 << (BitDepth_Y - 8)) * (219 * E'_R + 16)) \quad \text{(E-4)}$$

$$G = Clip1_Y((1 << (BitDepth_Y - 8)) * (219 * E'_G + 16)) \quad \text{(E-5)}$$

$$B = Clip1_Y((1 << (BitDepth_Y - 8)) * (219 * E'_B + 16)) \quad \text{(E-6)}$$

  – Otherwise, if matrix_coefficients is equal to 2, the interpretation of the matrix_coefficients syntax element is unknown or is determined by the application.

  – Otherwise (matrix_coefficients is not equal to 0, 1, 2, 4, 5, 6, 7, or 8), the interpretation of the matrix_coefficients syntax element is reserved for future definition by ITU-T | ISO/IEC.

– Otherwise (video_full_range_flag is equal to 1), the following equations apply.

  – If matrix_coefficients is equal to 1, 4, 5, 6, or 7, the following equations apply.

$$Y = Clip1_Y(Round(((1 << BitDepth_Y) - 1) * E'_Y)) \quad \text{(E-7)}$$

$$Cb = Clip1_C(Round(((1 << BitDepth_C) - 1) * E'_{PB} + (1 << (BitDepth_C - 1)))) \quad \text{(E-8)}$$

$$Cr = Clip1_C(Round(((1 << BitDepth_C) - 1) * E'_{PR} + (1 << (BitDepth_C - 1)))) \quad \text{(E-9)}$$

  – Otherwise, if matrix_coefficients is equal to 0 or 8, the following equations apply.

$$R = Clip1_Y(((1 << BitDepth_Y) - 1) * E'_R) \quad \text{(E-10)}$$

$$G = Clip1_Y(((1 << BitDepth_Y) - 1) * E'_G) \quad \text{(E-11)}$$

$$B = Clip1_Y(((1 << BitDepth_Y) - 1) * E'_B) \quad \text{(E-12)}$$

- Otherwise, if matrix_coefficients is equal to 2, the interpretation of the matrix_coefficients syntax element is unknown or is determined by the application.
- Otherwise (matrix_coefficients is not equal to 0, 1, 2, 4, 5, 6, 7, or 8), the interpretation of the matrix_coefficients syntax element is reserved for future definition by ITU-T | ISO/IEC.

– If matrix_coefficients is not equal to 0 or 8, the following equations apply.

$$E'_Y = K_R * E'_R + (1 - K_R - K_B) * E'_G + K_B * E'_B \quad \text{(E-13)}$$

$$E'_{PB} = 0.5 * (E'_B - E'_Y) \div (1 - K_B) \quad \text{(E-14)}$$

$$E'_{PR} = 0.5 * (E'_R - E'_Y) \div (1 - K_R) \quad \text{(E-15)}$$

NOTE 2 – E'_Y is analog with the value 0 associated with nominal black and the value 1 associated with nominal white. E'_{PB} and E'_{PR} are analog with the value 0 associated with both nominal black and nominal white. When transfer_characteristics is not equal to 11 or 12, E'_Y is analog with values in the range of 0 to 1. When transfer_characteristics is not equal to 11 or 12, E'_{PB} and E'_{PR} are analog with values in the range of –0.5 to 0.5. When transfer_characteristics is equal to 11 (IEC 61966-2-4), or 12 (ITU-R BT.1361 extended colour gamut system), E'_Y, E'_{PB} and E'_{PR} are analog with a larger range not specified in this Recommendation.

– Otherwise, if matrix_coefficients is equal to 0, the following equations apply.

$$Y = \text{Round}(G) \quad \text{(E-16)}$$

$$Cb = \text{Round}(B) \quad \text{(E-17)}$$

$$Cr = \text{Round}(R) \quad \text{(E-18)}$$

– Otherwise (matrix_coefficients is equal to 8), the following applies.
  – If BitDepth_C is equal to BitDepth_Y, the following equations apply.

$$Y = \text{Round}(0.5 * G + 0.25 * (R + B)) \quad \text{(E-19)}$$

$$Cb = \text{Round}(0.5 * G - 0.25 * (R + B)) + (1 << (\text{BitDepth}_C - 1)) \quad \text{(E-20)}$$

$$Cr = \text{Round}(0.5 * (R - B)) + (1 << (\text{BitDepth}_C - 1)) \quad \text{(E-21)}$$

NOTE 3 – For purposes of the YCgCo nomenclature used in Table E-5, Cb and Cr of Equations E-20 and E-21 may be referred to as Cg and Co, respectively. The inverse conversion for the above three equations should be computed as.

$$t = Y - (Cb - (1 << (\text{BitDepth}_C - 1))) \quad \text{(E-22)}$$

$$G = \text{Clip1}_Y(Y + (Cb - (1 << (\text{BitDepth}_C - 1)))) \quad \text{(E-23)}$$

$$B = \text{Clip1}_Y(t - (Cr - (1 << (\text{BitDepth}_C - 1)))) \quad \text{(E-24)}$$

$$R = \text{Clip1}_Y(t + (Cr - (1 << (\text{BitDepth}_C - 1)))) \quad \text{(E-25)}$$

  – Otherwise (BitDepth_C is not equal to BitDepth_Y), the following equations apply.

$$Cr = \text{Round}(R) - \text{Round}(B) + (1 << (\text{BitDepth}_C - 1)) \quad \text{(E-26)}$$

$$t = \text{Round}(B) + ((Cr - (1 << (\text{BitDepth}_C - 1))) >> 1) \quad \text{(E-27)}$$

$$Cb = Round(G) - t + (1 << (BitDepth_C - 1)) \quad \text{(E-28)}$$

$$Y = t + ((Cb - (1 << (BitDepth_C - 1))) >> 1) \quad \text{(E-29)}$$

NOTE 4 – For purposes of the YCgCo nomenclature used in Table E-5, Cb and Cr of Equations E-28 and E-26 may be referred to as Cg and Co, respectively. The inverse conversion for the above four equations should be computed as.

$$t = Y - ((Cb - (1 << (BitDepth_C - 1))) >> 1) \quad \text{(E-30)}$$

$$G = Clip1_Y(t + (Cb - (1 << (BitDepth_C - 1)))) \quad \text{(E-31)}$$

$$B = Clip1_Y(t - ((Cr - (1 << (BitDepth_C - 1))) >> 1)) \quad \text{(E-32)}$$

$$R = Clip1_Y(B + (Cr - (1 << (BitDepth_C - 1)))) \quad \text{(E-33)}$$

Table E-5 – Matrix coefficients

Value	Matrix	Informative remark
0	GBR	Typically referred to as RGB; see Equations E-16 to E-18
1	$K_R = 0.2126$; $K_B = 0.0722$	ITU-R Rec. BT.709-5 ITU-R Rec. BT.1361 conventional colour gamut system and extended colour gamut system IEC 61966-2-4 $xvYCC_{709}$ Society of Motion Picture and Television Engineers RP 177 (1993) Annex B
2	Unspecified	Image characteristics are unknown or are determined by the application.
3	Reserved	For future use by ITU-T | ISO/IEC
4	$K_R = 0.30$; $K_B = 0.11$	United States Federal Communications Commission Title 47 Code of Federal Regulations (2003) 73.682 (a) (20)
5	$K_R = 0.299$; $K_B = 0.114$	ITU-R Rec. BT.470-6 System B, G (historical) ITU-R Rec. BT.601-6 625 ITU-R Rec. BT.1358 625 ITU-R Rec. BT.1700 625 PAL and 625 SECAM IEC 61966-2-4 $xvYCC_{601}$ (functionally the same as the value 6)
6	$K_R = 0.299$; $K_B = 0.114$	ITU-R Rec. BT.601-6 525 ITU-R Rec. BT.1358 525 ITU-R Rec. BT.1700 NTSC Society of Motion Picture and Television Engineers 170M (2004) (functionally the same as the value 5)
7	$K_R = 0.212$; $K_B = 0.087$	Society of Motion Picture and Television Engineers 240M (1999)
8	YCgCo	See Equations E-19 to E-33
9-255	Reserved	For future use by ITU-T | ISO/IEC

chroma_loc_info_present_flag equal to 1 specifies that chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are present. chroma_loc_info_present_flag equal to 0 specifies that chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are not present.

chroma_sample_loc_type_top_field and **chroma_sample_loc_type_bottom_field** specify the location of chroma samples for the top field and the bottom field as shown in Figure E-1. The value of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field shall be in the range of 0 to 5, inclusive. When the

chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are not present, the values of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field shall be inferred to be equal to 0.

NOTE 5 – When coding progressive source material, chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should have the same value.

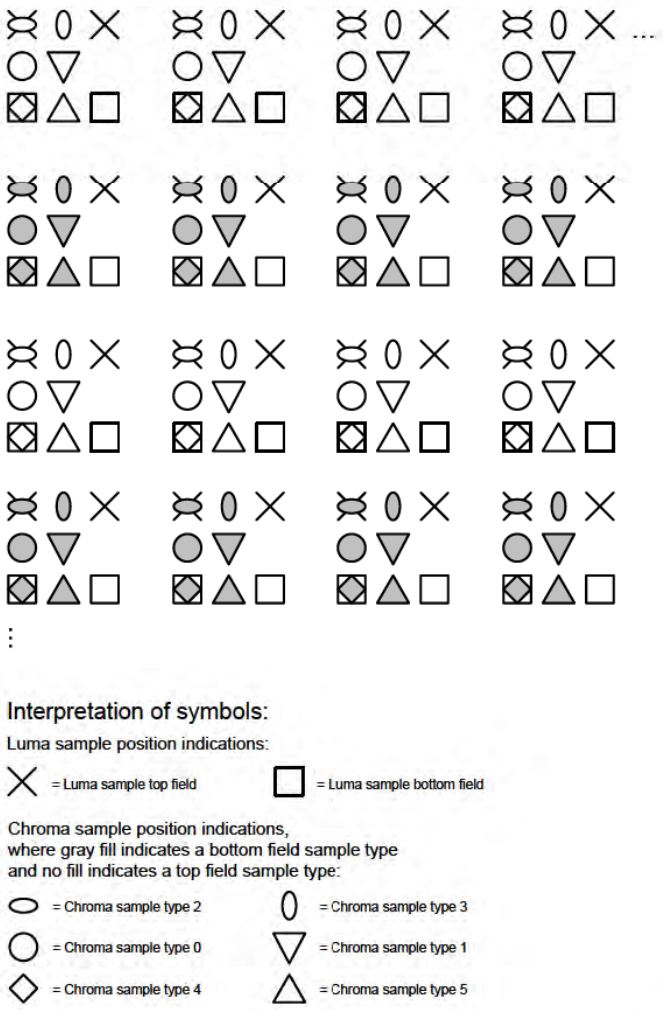


Figure E-1 – Location of chroma samples for top and bottom fields as a function of chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field

timing_info_present_flag equal to 1 specifies that num_units_in_tick, time_scale and fixed_frame_rate_flag are present in the bitstream. timing_info_present_flag equal to 0 specifies that num_units_in_tick, time_scale and fixed_frame_rate_flag are not present in the bitstream.

num_units_in_tick is the number of time units of a clock operating at the frequency time_scale Hz that corresponds to one increment (called a clock tick) of a clock tick counter. num_units_in_tick shall be greater than 0. A clock tick is the minimum interval of time that can be represented in the coded data. For example, when the clock frequency of a video signal is 60 000 ÷ 1001 Hz, time_scale may be equal to 60 000 and num_units_in_tick may be equal to 1001. See Equation C-1.

time_scale is the number of time units that pass in one second. For example, a time coordinate system that measures time using a 27 MHz clock has a time_scale of 27 000 000. time_scale shall be greater than 0.

fixed_frame_rate_flag equal to 1 indicates that the temporal distance between the HRD output times of any two consecutive pictures in output order is constrained as follows. fixed_frame_rate_flag equal to 0 indicates that no such constraints apply to the temporal distance between the HRD output times of any two consecutive pictures in output order.

For each picture n where n indicates the n-th picture (in output order) that is output and picture n is not the last picture in the bitstream (in output order) that is output, the value of $\Delta t_{fi,dpb}(n)$ is specified by

$$\Delta t_{fi,dpb}(n) = \Delta t_{o,dpb}(n) \div \text{DeltaTfiDivisor} \quad \text{(E-34)}$$

where $\Delta t_{o,dpb}(n)$ is specified in Equation C-13 and DeltaTfiDivisor is specified by Table E-6 based on the value of pic_struct_present_flag, field_pic_flag, and pic_struct for the coded video sequence containing picture n. Entries marked "-" in Table E-6 indicate a lack of dependence of DeltaTfiDivisor on the corresponding syntax element.

When fixed_frame_rate_flag is equal to 1 for a coded video sequence containing picture n, the value computed for $\Delta t_{fi,dpb}(n)$ shall be equal to t_c as specified in Equation C-1 (using the value of t_c for the coded video sequence containing picture n) when either or both of the following conditions are true for the following picture n_n that is specified for use in Equation C-13.

– picture n_n is in the same coded video sequence as picture n.

– picture n_n is in a different coded video sequence and fixed_frame_rate_flag is equal to 1 in the coded video sequence containing picture n_n and the value of num_units_in_tick ÷ time_scale is the same for both coded video sequences.

Table E-6 – Divisor for computation of $\Delta t_{fi,dpb}(n)$

pic_struct_present_flag	field_pic_flag	pic_struct	DeltaTfiDivisor
0	1	-	1
1	-	1	1
1	-	2	1
0	0	-	2
1	-	0	2
1	-	3	2
1	-	4	2
1	-	5	3
1	-	6	3
1	-	7	4
1	-	8	6

nal_hrd_parameters_present_flag equal to 1 specifies that NAL HRD parameters (pertaining to Type II bitstream conformance) are present. nal_hrd_parameters_present_flag equal to 0 specifies that NAL HRD parameters are not present.

NOTE 6 – When nal_hrd_parameters_present_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the NAL HRD parameters, including the NAL sequence HRD parameter information and all buffering period and picture timing SEI messages, by some means not specified in this Recommendation | International Standard.

When nal_hrd_parameters_present_flag is equal to 1, NAL HRD parameters (subclauses E.1.2 and E.2.2) immediately follow the flag.

The variable NalHrdBpPresentFlag is derived as follows.

– If any of the following is true, the value of NalHrdBpPresentFlag shall be set equal to 1.
  – nal_hrd_parameters_present_flag is present in the bitstream and is equal to 1,
  – the need for presence of buffering periods for NAL HRD operation to be present in the bitstream in buffering period SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.

- Otherwise, the value of NalHrdBpPresentFlag shall be set equal to 0.

vcl_hrd_parameters_present_flag equal to 1 specifies that VCL HRD parameters (pertaining to all bitstream conformance) are present. vcl_hrd_parameters_present_flag equal to 0 specifies that VCL HRD parameters are not present.

NOTE 7 – When vcl_hrd_parameters_present_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the VCL HRD parameters and all buffering period and picture timing SEI messages, by some means not specified in this Recommendation | International Standard.

When vcl_hrd_parameters_present_flag is equal to 1, VCL HRD parameters (subclauses E.1.2 and E.2.2) immediately follow the flag.

The variable VclHrdBpPresentFlag is derived as follows.

– If any of the following is true, the value of VclHrdBpPresentFlag shall be set equal to 1.
  – vcl_hrd_parameters_present_flag is present in the bitstream and is equal to 1,
  – the need for presence of buffering periods for VCL HRD operation to be present in the bitstream in buffering period SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.
– Otherwise, the value of VclHrdBpPresentFlag shall be set equal to 0.

The variable CpbDpbDelaysPresentFlag is derived as follows.

– If any of the following is true, the value of CpbDpbDelaysPresentFlag shall be set equal to 1.
  – nal_hrd_parameters_present_flag is present in the bitstream and is equal to 1,
  – vcl_hrd_parameters_present_flag is present in the bitstream and is equal to 1,
  – the need for presence of CPB and DPB output delays to be present in the bitstream in picture timing SEI messages is determined by the application, by some means not specified in this Recommendation | International Standard.
– Otherwise, the value of CpbDpbDelaysPresentFlag shall be set equal to 0.

low_delay_hrd_flag specifies the HRD operational mode as specified in Annex C. When fixed_frame_rate_flag is equal to 1, low_delay_hrd_flag shall be equal to 0.

NOTE 8 – When low_delay_hrd_flag is equal to 1, "big pictures" that violate the nominal CPB removal times due to the number of bits used by an access unit are permitted. It is expected, but not required, that such "big pictures" occur only occasionally.

pic_struct_present_flag equal to 1 specifies that picture timing SEI messages (subclause D.2.2) are present that include the pic_struct syntax element. pic_struct_present_flag equal to 0 specifies that the pic_struct syntax element is not present in picture timing SEI messages. When pic_struct_present_flag is not present, its value shall be inferred to be equal to 0.

bitstream_restriction_flag equal to 1, specifies that the following coded video sequence bitstream restriction parameters are present. bitstream_restriction_flag equal to 0, specifies that the following coded video sequence bitstream restriction parameters are not present.

motion_vectors_over_pic_boundaries_flag equal to 0 indicates that no sample outside the picture boundaries and no sample at a fractional sample position whose value is derived using one or more samples outside the picture boundaries is used to inter predict any sample. motion_vectors_over_pic_boundaries_flag equal to 1 indicates that one or more samples outside picture boundaries may be used in inter prediction. When the motion_vectors_over_pic_boundaries_flag syntax element is not present, motion_vectors_over_pic_boundaries_flag value shall be inferred to be equal to 1.

max_bytes_per_pic_denom indicates a number of bytes not exceeded by the sum of the sizes of the VCL NAL units associated with any coded picture in the coded video sequence.

The number of bytes that represent a picture in the NAL unit stream is specified for this purpose as the total number of bytes of VCL NAL unit data (i.e., the total of the NumBytesInNALunit variables for the VCL NAL units) for the picture. The value of max_bytes_per_pic_denom shall be in the range of 0 to 16, inclusive.

Depending on max_bytes_per_pic_denom the following applies.

– If max_bytes_per_pic_denom is equal to 0, no limits are indicated.
– Otherwise (max_bytes_per_pic_denom is not equal to 0), no coded picture shall be represented in the coded video sequence by more than the following number of bytes.

$$(PicSizeInMbs * RawMbBits) \div (8 * max\_bytes\_per\_pic\_denom) \quad \text{(E-35)}$$

When the max_bytes_per_pic_denom syntax element is not present, the value of max_bytes_per_pic_denom shall be inferred to be equal to 2.

max_bits_per_mb_denom indicates the maximum number of coded bits of macroblock_layer( ) data for any macroblock in any picture of the coded video sequence. The value of max_bits_per_mb_denom shall be in the range of 0 to 16, inclusive.

Depending on max_bits_per_mb_denom the following applies.

– If max_bits_per_mb_denom is equal to 0, no limit is specified.

– Otherwise (max_bits_per_mb_denom is not equal to 0), no coded macroblock_layer( ) shall be represented in the bitstream by more than the following number of bits.

$$(128 + \text{RawMbBits}) \div \text{max\_bits\_per\_mb\_denom} \quad \text{(E-36)}$$

Depending on entropy_coding_mode_flag, the bits of macroblock_layer( ) data are counted as follows.

– If entropy_coding_mode_flag is equal to 0, the number of bits of macroblock_layer( ) data is given by the number of bits in the macroblock_layer( ) syntax structure for a macroblock.

– Otherwise (entropy_coding_mode_flag is equal to 1), the number of bits of macroblock_layer( ) data for a macroblock is given by the number of times read_bits(1) is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing the macroblock_layer( ) associated with the macroblock.

When the max_bits_per_mb_denom is not present, the value of max_bits_per_mb_denom shall be inferred to be equal to 1.

log2_max_mv_length_horizontal and **log2_max_mv_length_vertical** indicate the maximum absolute value of a decoded horizontal and vertical motion vector component, respectively, in ¼ luma sample units, for all pictures in the coded video sequence. A value of n asserts that no value of a motion vector component shall exceed the range from -2^n to 2^n-1, inclusive, in units of ¼ luma sample displacement. The value of log2_max_mv_length_horizontal shall be in the range of 0 to 16, inclusive. The value of log2_max_mv_length_vertical shall be in the range of 0 to 16, inclusive. When log2_max_mv_length_horizontal is not present, the values of log2_max_mv_length_horizontal and log2_max_mv_length_vertical shall be inferred to be equal to 16.

NOTE 9 – The maximum absolute value of a decoded vertical or horizontal motion vector component is also constrained by profile and level limits as specified in Annex A.

num_reorder_frames indicates the maximum number of frames, complementary field pairs, or non-paired fields that precede any frame, complementary field pair, or non-paired field in the coded video sequence in decoding order and follow it in output order. The value of num_reorder_frames shall be in the range of 0 to max_dec_frame_buffering, inclusive. When the num_reorder_frames syntax element is not present, the value of num_reorder_frames value shall be inferred as follows.

– If profile_idc is equal to 44, 86, 100, 110, 122, or 244 and constraint_set3_flag is equal to 1, the value of num_reorder_frames shall be inferred to be equal to 0.

– Otherwise (profile_idc is not equal to 44, 86, 100, 110, 122, or 244 or constraint_set3_flag is equal to 0), the value of num_reorder_frames shall be inferred to be equal to MaxDpbSize.

max_dec_frame_buffering specifies the required size of the HRD decoded picture buffer (DPB) in units of frame buffers. The coded video sequence shall not require a decoded picture buffer with size of more than Max(1, max_dec_frame_buffering) frame buffers to enable the output of decoded pictures at the output times specified by dpb_output_delay of the picture timing SEI messages. The value of max_dec_frame_buffering shall be in the range of num_ref_frames to MaxDpbSize (as specified in subclause A.3.1 or A.3.2), inclusive. When the max_dec_frame_buffering syntax element is not present, the value of max_dec_frame_buffering shall be inferred as follows.

– If profile_idc is equal to 44 or 244 and constraint_set3_flag is equal to 1, the value of max_dec_frame_buffering shall be inferred to be equal to 0.

– Otherwise (profile_idc is not equal to 44 or 244 or constraint_set3_flag is equal to 0), the value of max_dec_frame_buffering shall be inferred to be equal to MaxDpbSize.

E.2.2 HRD parameters semantics

cpb_cnt_minus1 plus 1 specifies the number of alternative CPB specifications in the bitstream. The value of cpb_cnt_minus1 shall be in the range of 0 to 31, inclusive. When low_delay_hrd_flag is equal to 1, cpb_cnt_minus1 shall be equal to 0. When cpb_cnt_minus1 is not present, it shall be inferred to be equal to 0.

bit_rate_scale (together with bit_rate_value_minus1[SchedSelIdx]) specifies the maximum input bit rate of the SchedSelIdx-th CPB.

cpb_size_scale (together with cpb_size_value_minus1[SchedSelIdx]) specifies the CPB size of the SchedSelIdx-th CPB.

bit_rate_value_minus1[SchedSelIdx **]** (together with bit_rate_scale) specifies the maximum input bit rate for the SchedSelIdx-th CPB. bit_rate_value_minus1[SchedSelIdx] shall be in the range of 0 to $2^{32} - 2$, inclusive. For any SchedSelIdx > 0, bit_rate_value_minus1[SchedSelIdx] shall be greater than bit_rate_value_minus1[SchedSelIdx - 1]. The bit rate in bits per second is given by

$$BitRate[SchedSelIdx] = (bit\_rate\_value\_minus1[SchedSelIdx] + 1) * 2^{(6 + bit\_rate\_scale)} \quad (E\text{-}37)$$

When the bit_rate_value_minus1[SchedSelIdx] syntax element is not present, the value of BitRate[SchedSelIdx] shall be inferred as follows.

- If profile_idc is equal to 66, 77, or 88, BitRate[SchedSelIdx] shall be inferred to be equal to 1000 * MaxBR for VCL HRD parameters and to be equal to 1200 * MaxBR for NAL HRD parameters, where MaxBR is specified in subclause A.3.1.
- Otherwise, BitRate[SchedSelIdx] shall be inferred to be equal to cpbBrVclFactor * MaxBR for VCL HRD parameters and to be equal to cpbBrNalFactor * MaxBR for NAL HRD parameters, where cpbBrVclFactor, cpbBrNalFactor, and MaxBR are specified in subclause A.3.3.

cpb_size_value_minus1[SchedSelIdx **]** is used together with cpb_size_scale to specify the SchedSelIdx-th CPB size. cpb_size_value_minus1[SchedSelIdx] shall be in the range of 0 to $2^{32} - 2$, inclusive. For any SchedSelIdx greater than 0, cpb_size_value_minus1[SchedSelIdx] shall be less than or equal to cpb_size_value_minus1[SchedSelIdx -1].

The CPB size in bits is given by

$$CpbSize[SchedSelIdx] = (cpb\_size\_value\_minus1[SchedSelIdx] + 1) * 2^{(4 + cpb\_size\_scale)} \quad (E\text{-}38)$$

When the cpb_size_value_minus1[SchedSelIdx] syntax element is not present, the value of CpbSize[SchedSelIdx] shall be inferred as follows.

- If profile_idc is equal to 66, 77, or 88, CpbSize[SchedSelIdx] shall be inferred to be equal to 1000 * MaxCPB for VCL HRD parameters and to be equal to 1200 * MaxCPB for NAL HRD parameters, where MaxCPB is specified in subclause A.3.1.
- Otherwise, CpbSize[SchedSelIdx] shall be inferred to be equal to cpbBrVclFactor * MaxCPB for VCL HRD parameters and to be equal to cpbBrNalFactor * MaxCPB for NAL HRD parameters, where cpbBrVclFactor, cpbBrNalFactor, and MaxCPB are specified in subclause A.3.3.

cbr_flag[SchedSelIdx **]** equal to 0 specifies that to decode this bitstream by the HRD using the SchedSelIdx-th CPB specification, the hypothetical stream delivery scheduler (HSS) operates in an intermittent bit rate mode. cbr_flag[SchedSelIdx] equal to 1 specifies that the HSS operates in a constant bit rate (CBR) mode. When the cbr_flag[SchedSelIdx] syntax element is not present, the value of cbr_flag shall be inferred to be equal to 0.

initial_cpb_removal_delay_length_minus1 specifies the length in bits of the initial_cpb_removal_delay[SchedSelIdx] and initial_cpb_removal_delay_offset[SchedSelIdx] syntax elements of the buffering period SEI message. The length of initial_cpb_removal_delay[SchedSelIdx] and of initial_cpb_removal_delay_offset[SchedSelIdx] is initial_cpb_removal_delay_length_minus1 + 1. When the initial_cpb_removal_delay_length_minus1 syntax element is present in more than one hrd_parameters( ) syntax structure within the VUI parameters syntax structure, the value of the initial_cpb_removal_delay_length_minus1 parameters shall be equal in both hrd_parameters( ) syntax structures. When the initial_cpb_removal_delay_length_minus1 syntax element is not present, it shall be inferred to be equal to 23.

cpb_removal_delay_length_minus1 specifies the length in bits of the cpb_removal_delay syntax element. The length of the cpb_removal_delay syntax element of the picture timing SEI message is cpb_removal_delay_length_minus1 + 1. When the cpb_removal_delay_length_minus1 syntax element is present in more than one hrd_parameters( ) syntax structure within the VUI parameters syntax structure, the value of the cpb_removal_delay_length_minus1 parameters

shall be equal in both hrd_parameters( ) syntax structures. When the cpb_removal_delay_length_minus1 syntax element is not present, it shall be inferred to be equal to 23.

dpb_output_delay_length_minus1 specifies the length in bits of the dpb_output_delay syntax element. The length of the dpb_output_delay syntax element of the picture timing SEI message is dpb_output_delay_length_minus1 + 1. When the dpb_output_delay_length_minus1 syntax element is present in more than one hrd_parameters( ) syntax structure within the VUI parameters syntax structure, the value of the dpb_output_delay_length_minus1 parameters shall be equal in both hrd_parameters( ) syntax structures. When the dpb_output_delay_length_minus1 syntax element is not present, it shall be inferred to be equal to 23.

time_offset_length greater than 0 specifies the length in bits of the time_offset syntax element. time_offset_length equal to 0 specifies that the time_offset syntax element is not present. When the time_offset_length syntax element is present in more than one hrd_parameters( ) syntax structure within the VUI parameters syntax structure, the value of the time_offset_length parameters shall be equal in both hrd_parameters( ) syntax structures. When the time_offset_length syntax element is not present, it shall be inferred to be equal to 24.

Annex G
Scalable video coding

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

This annex specifies scalable video coding, referred to as SVC.

G.1 Scope

Bitstreams and decoders conforming to one or more of the profiles specified in this annex are completely specified in this annex with reference made to clauses 2-9 and Annexes A-E.

G.2 Normative References

The specifications in clause 2 apply with the following additions.

- ISO/IEC 10646:2003, Information technology – Universal Multiple-Octet Coded Character Set (UCS).
- IETF RFC 3986 (2005), Uniform Resource Identifiers (URI): Generic Syntax.

G.3 Definitions

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

G.3.1 arbitrary slice order (ASO): A *decoding order* of *slices* in which the *macroblock address* of the first *macroblock* of some *slice* of a *slice group* within a *layer representation* may be less than the *macroblock address* of the first *macroblock* of some other preceding *slice* of the same *slice group* within the same *layer representation* or in which the *slices* of a *slice group* within a *layer representation* may be interleaved with the *slices* of one or more other *slices groups* within the same *layer representation*.

G.3.2 associated NAL unit: A *NAL unit* that directly succeeds a *prefix NAL unit* in *decoding order*.

G.3.3 B slice: A *slice* that may be decoded using *intra-layer intra prediction* or *inter prediction* using at most two *motion vectors* and *reference indices* to *predict* the sample values of each *block*.

G.3.4 base layer: A *bitstream subset* that contains all the *NAL units* with the nal_unit_type *syntax element* equal to 1 and 5 of the *bitstream* and does not contain any *NAL unit* with the nal_unit_type *syntax element* equal to 14, 15, or 20 and conforms to one or more of the profiles specified in Annex A.

G.3.5 base quality layer representation: The *layer representation* of the *target dependency representation* of an *access unit* that is associated with the quality_id *syntax element* equal to 0.

G.3.6 bitstream subset: A *bitstream* that is derived as a *subset* from a *bitstream* by discarding zero or more *NAL units*. A *bitstream subset* is also referred to as *sub-bitstream*.

G.3.7 bottom macroblock (of a macroblock pair): The *macroblock* within a *macroblock pair* that contains the samples in the bottom row of samples for the *macroblock pair*. For a *field macroblock pair*, the bottom macroblock represents the samples from the region of the *bottom field* or *layer bottom field* of the *frame* or *layer frame*, respectively, that lie within the spatial region of the *macroblock pair*. For a *frame macroblock pair*, the bottom macroblock represents the samples of the *frame* or *layer frame* that lie within the bottom half of the spatial region of the *macroblock pair*.

G.3.8 coded slice in scalable extension NAL unit: A *coded slice NAL unit* that contains an *EI slice*, *EP slice*, or an *EB slice*.

G.3.9 complementary reference field pair: A collective term for two *reference fields* that are in consecutive *access units* in *decoding order* as two *coded fields*, where the *target dependency representations* of the *access units* share the same value of the frame_num *syntax element* and where the second *field* in *decoding order* is not an *IDR picture* and the *target dependency representation* of the second *access unit* does not include a memory_management_control_operation *syntax element* equal to 5, or a *complementary reference base field pair*.

G.3.10 complementary reference base field pair: Two *reference base fields* that are constructed from consecutive *access units* in *decoding order*, where the *target dependency representations* of the *access units* share the same value of the frame_num *syntax element* and where the second *access unit* in *decoding order* is not an *IDR*

access unit and the *target dependency representation* of the second *access unit* does not include a memory_management_control_operation *syntax element* equal to 5. A *complementary reference base field pair* is a *complementary reference field pair*.

G.3.11 dependency representation: A *subset* of *VCL NAL units* within an *access unit* that are associated with the same value of the dependency_id *syntax element*, which is provided as part of the *NAL unit* header or by an associated *prefix NAL unit*. A *dependency representation* consist of one or more *layer representations*.

G.3.12 EB slice: A *slice* that may be decoded using *intra prediction* or *inter prediction* or *inter-layer prediction* from *syntax elements* and derived variables of the *reference layer representation*. For *inter-prediction* of EB slices at most two *motion vectors* and *reference indices* are used to *predict* the sample values of each *block*.

G.3.13 EI slice: A *slice* that is not an *I slice* or *SI slice* that is decoded using *intra prediction* only.

G.3.14 EP slice: A *slice* that may be decoded using *intra prediction* or *inter prediction* or *inter-layer prediction* from *syntax elements* and derived variables of the *reference layer representation*. For *inter-prediction* of EP slices at most one *motion vector* and *reference index* is used to *predict* the sample values of each *block*.

G.3.15 field macroblock: A *macroblock* containing samples from a single *field* or *layer field*.

G.3.16 filler prefix NAL unit: A *prefix NAL unit* with an *associated NAL unit* that is a filler data *NAL unit*.

G.3.17 frame macroblock: A *macroblock* containing samples from the two *fields* or *layer fields* of a *frame* or *layer frame*, respectively.

G.3.18 I slice: A *slice* that is decoded using *intra-layer intra prediction* only.

G.3.19 instantaneous decoding refresh (IDR) picture: A *coded picture* in which all *slices* of the *target dependency representation* within the access unit are *I or EI slices* that causes the *decoding process* to mark all *reference pictures* as "unused for reference" immediately after decoding the IDR picture. After the decoding of an IDR picture all following *coded pictures* in *decoding order* can be decoded without *inter prediction* from any *picture* decoded prior to the IDR picture. The first *picture* of each *coded video sequence* is an IDR picture.

G.3.20 inter-layer intra prediction: An *inter-layer prediction* derived from decoded samples of *intra-coded macroblocks* of the *reference layer representation*.

G.3.21 inter-layer prediction: A *prediction* derived from *syntax elements*, derived variables, or decoded samples of the *reference layer representation*.

G.3.22 intra-layer intra prediction: A *prediction* derived from decoded samples of the same decoded *slice*.

G.3.23 intra prediction: A collective term for *intra-layer intra prediction* or *inter-layer intra prediction* or a combination of *intra-layer intra prediction* together with *inter-layer prediction* from *syntax elements* and derived variables of the *reference layer representation*.

G.3.24 intra slice: A collective term for *I slice* or *EI slice*.

G.3.25 layer bottom field: One of two *layer fields* that comprise a *layer frame*. Each row of a layer bottom field is spatially located immediately below a corresponding row of a *layer top field*.

G.3.26 layer field: An assembly of alternate rows of a *layer frame*. A *layer frame* is composed of two *layer fields*, a *layer top field* and a *layer bottom field*.

G.3.27 layer frame: A *layer frame* contains an array of *luma* samples that represents an intermediate decoding result for a *field* or a *frame* in monochrome format or an array of *luma* samples and two corresponding arrays of *chroma* samples that represent an intermediate decoding result for a *field* or a *frame* in 4:2:0, 4:2:2, and 4:4:4 colour format. A *layer frame* consists of two *layer fields*, a *layer top field* and a *layer bottom field*.

G.3.28 layer picture: A collective term for a *layer field* or a *layer frame*.

G.3.29 layer top field: One of two *layer fields* that comprise a *layer frame*. Each row of a *layer top field* is spatially located immediately above a corresponding row of a *layer bottom field*.

G.3.30 layer representation: A subset of *VCL NAL units* within an *access unit* that are associated with the same values of the dependency_id and quality_id *syntax elements*, which are provided as part of the *VCL NAL unit* header or by an associated *prefix NAL unit*. One or more *layer representations* represent a *dependency representation*.

G.3.31 layer representation identifier: An integer value by which a particular *layer representation* inside an *access unit* is uniquely identified.

G.3.32 **macroblock**: A 16x16 *block* of *luma* samples and two corresponding *blocks* of *chroma* samples of a *picture* or *layer picture* that has three sample arrays, or a 16x16 *block* of samples of a monochrome *picture* or *layer picture*. The division of a *slice* or a *macroblock pair* into macroblocks is a *partitioning*.

G.3.33 **macroblock-adaptive frame/field decoding**: A *decoding process* for *coded frames* or *layer representations* in which some *macroblocks* may be decoded as *frame macroblocks* and others may be decoded as *field macroblocks*.

G.3.34 **macroblock address**: When *macroblock-adaptive frame/field decoding* is not in use, a macroblock address is the index of a macroblock in a *macroblock raster scan* of the *picture* or *layer picture* starting with zero for the top-left *macroblock* in a *picture* or *layer picture*. When *macroblock-adaptive frame/field decoding* is in use, the macroblock address of the *top macroblock* of a *macroblock pair* is two times the index of the *macroblock pair* in a *macroblock pair raster scan* of the *picture* or *layer picture*, and the macroblock address of the *bottom macroblock* of a *macroblock pair* is the macroblock address of the corresponding *top macroblock* plus 1. The macroblock address of the *top macroblock* of each *macroblock pair* is an even number and the macroblock address of the *bottom macroblock* of each *macroblock pair* is an odd number.

G.3.35 **macroblock location**: The two-dimensional coordinates of a *macroblock* in a *picture* or *layer picture* denoted by (x, y). For the top left *macroblock* of the *picture* or *layer picture* (x, y) is equal to (0, 0). x is incremented by 1 for each *macroblock* column from left to right. When *macroblock-adaptive frame/field decoding* is not in use, y is incremented by 1 for each *macroblock* row from top to bottom. When *macroblock-adaptive frame/field decoding* is in use, y is incremented by 2 for each *macroblock pair* row from top to bottom, and is incremented by an additional 1 when a macroblock is a *bottom macroblock*.

G.3.36 **macroblock pair**: A pair of vertically contiguous *macroblocks* in a *frame* or *layer frame* that is coupled for use in *macroblock-adaptive frame/field decoding*. The division of a *slice* into macroblock pairs is a *partitioning*.

G.3.37 **macroblock to slice group map**: A means of mapping *macroblocks* of a *picture* or *layer picture* into *slice groups*. The macroblock to slice group map consists of a list of numbers, one for each coded *macroblock*, specifying the *slice group* to which each coded *macroblock* belongs.

G.3.38 **map unit to slice group map**: A means of mapping *slice group map units* of a *picture* or *layer picture* into *slice groups*. The map unit to slice group map consists of a list of numbers, one for each *slice group map unit*, specifying the *slice group* to which each coded *slice group map unit* belongs to.

G.3.39 **non-paired reference base field**: A *reference base field* that is not part of a *complementary reference base field pair*. A *non-paired reference base field* is a *non-paired reference field*.

G.3.40 **P slice**: A *slice* that may be decoded using *intra-layer intra prediction* or inter *prediction* using at most one *motion vector* and *reference index* to *predict* the sample values of each *block*.

G.3.41 **parameter**: A *syntax element* of an *SVC sequence parameter set* or a *picture parameter set*. Parameter is also used as part of the defined term *quantisation parameter*.

G.3.42 **picture parameter set**: A *syntax structure* containing *syntax elements* that apply to zero or more *layer representations* as determined by the pic_parameter_set_id *syntax element* found in each *slice header*.

G.3.43 **prefix NAL unit**: A *NAL unit* with nal_unit_type equal to 14 that immediately precedes in *decoding order* a *NAL unit* with nal_unit_type equal to 1, 5, or 12. The *NAL unit* that immediately succeeds in *decoding order* the *prefix NAL unit* is referred to as the *associated NAL unit*. The *prefix NAL unit* contains data associated with the *associated NAL unit*, which are considered to be part of the *associated NAL unit*.

G.3.44 **reference base field**: A *reference field* that is obtained by decoding a *base quality layer representation* with the nal_ref_idc *syntax element* not equal to 0, the store_ref_base_pic_flag *syntax element* equal to 1, and the field_pic_flag *syntax element* equal to 1 of an access unit and all *layer representations* of the *access unit* that are referred to by *inter-layer prediction* of the *base quality layer representation*. A *reference base field* is not an output of the *decoding process*, but may be used for *inter prediction* when *P*, *B*, *EP*, and *EB slices* of a *coded field* or *field macroblock* of a *coded frame* are decoded. See also *reference base picture*.

G.3.45 **reference base frame**: A *reference frame* that is obtained by decoding a *base quality layer representation* with the nal_ref_idc *syntax element* not equal to 0, the store_ref_base_pic_flag *syntax element* equal to 1, and the field_pic_flag *syntax element* equal to 0 of an access unit and all *layer representations* of the *access unit* that are referred to by *inter-layer prediction* of the *base quality layer representation*. A *reference base frame* is not an output of the *decoding process*, but may be used for *inter prediction* when *P*, *B*, *EP*, and *EB slices* of a *coded frame* are decoded. See also *reference base picture*.

G.3.46 **reference base picture**: A *reference picture* that is obtained by decoding a *base quality layer representation* with the nal_ref_idc *syntax element* not equal to 0 and the store_ref_base_pic_flag *syntax element* equal to 1 of

an access unit and all *layer representations* of the *access unit* that are referred to by *inter-layer prediction* of the *base quality layer representation*. A *reference base picture* is not an output of the *decoding process*, but the samples of a *reference base picture* may be used for *inter prediction* in the *decoding process* of subsequent *pictures* in *decoding order*. *Reference base picture* is a collective term for a *reference base field* or a *reference base frame*.

G.3.47 **reference field**: A *reference field* may be used for *inter prediction* when *P, B, EP,* or *EB slices* of a *coded field* or *field macroblocks* of a *coded frame* are decoded. See also *reference picture*.

G.3.48 **reference frame**: A *reference frame* may be used for *inter prediction* when *P, B, EP,* or *EB slices* of a *coded frame* are decoded. See also *reference picture*.

G.3.49 **reference layer macroblock**: A *macroblock* of a *reference layer representation*.

G.3.50 **reference layer representation**: A *reference layer representation* for a particular *layer representation* of an *access unit* is the *layer representation* that is used for *inter-layer prediction* of the particular *layer representation*. The *reference layer representation* is in the same *access unit* as the *layer representation* that uses the *reference layer representation* for *inter-layer prediction*.

G.3.51 **reference picture**: A collective term for a *decoded picture* that is obtained by decoding an *access unit* for that the nal_ref_idc *syntax element* that is associated with the *target dependency representation* is not equal to 0 or a *reference base picture*. A *reference picture* contains samples that may be used for *inter prediction* in the *decoding process* of subsequent *pictures* in *decoding order*.

G.3.52 **reference picture list**: A list of *reference pictures* that is used for *inter prediction* of a *P, B, EP,* or *EB slice*. For the *decoding process* of a *P* or *EP slice,* there is one reference picture list. For the *decoding process* of a *B* or *EB slice*, there are two reference picture lists.

G.3.53 **reference picture list 0**: A *reference picture list* used for *inter prediction* of a *P, B, EP,* or *EB slice*. All *inter prediction* used for *P* and *EP slices* uses reference picture list 0. Reference picture list 0 is one of two *reference picture lists* used for *inter prediction* for a *B* or *EB slice*, with the other being *reference picture list 1*.

G.3.54 **reference picture list 1**: A *reference picture list* used for *inter prediction* of a *B* or *EB slice*. Reference picture list 1 is one of two *reference picture lists* used for *inter prediction* for a *B* or *EB slice*, with the other being *reference picture list 0*.

G.3.55 **scalable bitstream**: A *bitstream* with the property that one or more *bitstream subsets* that are not identical to the *scalable bitstream* form another *bitstream* that conforms to this specification.

G.3.56 **sequence parameter set**: A *syntax structure* containing *syntax elements* that apply to zero or more *layer representations* with the dependency_id *syntax element* equal to 0 and the quality_id *syntax element* equal to 0 as determined by the content of a seq_parameter_set_id *syntax element* found in the *picture parameter set* referred to by the pic_parameter_set_id *syntax element* found in each *slice header* of *I*, *P*, and *B slices*.

G.3.57 **slice**: An integer number of *macroblocks* or *macroblock pairs* ordered consecutively in the *raster scan* within a particular *slice group*. For a *picture* or *layer picture*, the division of each *slice group* into slices is a partitioning. Although a slice contains *macroblocks* or *macroblock pairs* that are consecutive in the *raster scan* within a *slice group*, these *macroblocks* or *macroblock pairs* are not necessarily consecutive in the *raster scan* within the *picture* or *layer picture*. The *macroblock addresses* are derived from the first *macroblock address* in a slice (as represented in the *slice header*) and the *macroblock to slice group map*.

G.3.58 **slice group**: A subset of the *macroblocks* or *macroblock pairs* of a *picture* or *layer picture*. The division of the *picture* or *layer picture* into slice groups is a *partitioning* of the *picture* or *layer picture*. The partitioning is specified by the *macroblock to slice group map*.

G.3.59 **spatial intra prediction**: See *intra-layer intra prediction*.

G.3.60 **sub-bitstream**: A *subset* of a *bitstream*. A *sub-bitstream* is also referred to as *bitstream subset*.

G.3.61 **subset**: A subset contains only elements that are also contained in the set from which the subset is derived. The subset may be identical to the set from which it is derived.

G.3.62 **subset sequence parameter set**: A *syntax structure* containing *syntax elements* that apply to zero or more *layer representations* with the dependency_id *syntax element* not equal to 0 or the quality_id *syntax element* not equal to 0 as determined by the content of a seq_parameter_set_id *syntax element* found in the *picture parameter set* referred to by the pic_parameter_set_id *syntax element* found in each *slice header* of *EI*, *EP*, and *EB slices*.

G.3.63 **SVC sequence parameter set**: A collective term for *sequence parameter set* or *subset sequence parameter set*.

G.3.64 SVC sequence parameter set RBSP: A collective term for sequence parameter set RBSP or subset sequence parameter set RBSP.

G.3.65 target dependency representation: The *dependency representation* of an *access unit* that is associated with the largest value of the dependency_id *syntax element* for all *dependency representations* of the *access unit*.

G.3.66 target layer representation: The *layer representation* of the *target dependency representation* of an *access unit* that is associated with the largest value of the quality_id *syntax element* for all *layer representations* of the *target dependency representation* of the *access unit*.

G.3.67 top macroblock (of a macroblock pair): The *macroblock* within a *macroblock pair* that contains the samples in the top row of samples for the *macroblock pair*. For a *field macroblock pair*, the top macroblock represents the samples from the region of the *top field* or *layer top field* of the *frame* or *layer frame* that lie within the spatial region of the *macroblock pair*. For a *frame macroblock pair*, the top macroblock represents the samples of the *frame* or *layer frame* that lie within the top half of the spatial region of the *macroblock pair*.

G.3.68 VCL NAL unit: A collective term for *coded slice NAL units* and *VCL prefix NAL units*.

G.3.69 VCL prefix NAL unit: A *prefix NAL unit* with an *associated NAL unit* that has nal_unit_type equal to 1 or 5.

G.4 Abbreviations

The specifications in clause 4 apply.

G.5 Conventions

The specifications in clause 5 apply.

G.6 Source, coded, decoded and output data formats, scanning processes, neighbouring and reference layer relationships

The specifications in clause 6 apply with substituting SVC sequence parameter set for sequence parameter set. Additionally, the following process is specified.

G.6.1 Derivation process for reference layer macroblocks

This process is only invoked when no_inter_layer_pred_flag is equal to 0.

Inputs to this process are

- a luma location (xP, yP) relative to the upper-left luma sample of the current macroblock,
- a variable fieldMbFlag specifying whether the current macroblock is a field or a frame macroblock,
- a one-dimensional array refLayerFieldMbFlag with RefLayerPicSizeInMbs elements specifying which macroblocks of the reference layer representation are field macroblocks and which macroblocks are frame macroblocks,
- a one-dimensional array refLayerMbType with RefLayerPicSizeInMbs elements specifying macroblock types for the macroblocks of the reference layer representation.

Outputs of this process are

- the macroblock address mbAddrRefLayer specifying the reference layer macroblock,
- a luma location (xB, yB) relative to the upper-left luma sample of the reference layer macroblock.

The variables shiftX, shiftY, scaleX, scaleY, offsetX, and offsetY are derived as specified in the following ordered steps.

NOTE 1 – These variables do not depend on the luma location (xP, yP), the variable fieldMbFlag, or the current macroblock.

1. The variables refW, refH, scaledW, scaledH, offsetX, and offsetY are derived by

$$refW = RefLayerPicWidthInSamples_L \quad (G\text{-}1)$$
$$refH = RefLayerPicHeightInSamples_L \quad (G\text{-}2)$$
$$scaledW = ScaledRefLayerPicWidthInSamples_L \quad (G\text{-}3)$$
$$scaledH = ScaledRefLayerPicHeightInSamples_L \quad (G\text{-}4)$$
$$offsetX = ScaledRefLayerLeftOffset \quad (G\text{-}5)$$
$$offsetY = ScaledRefLayerTopOffset / (1 + field\_pic\_flag) \quad (G\text{-}6)$$

2. The variables shiftX and shiftY are derived by

$$shiftX = ((level\_idc <= 30) \ ? \ 16 \ : \ (31 - Ceil(Log2(refW)))) \quad (G\text{-}7)$$
$$shiftY = ((level\_idc <= 30) \ ? \ 16 \ : \ (31 - Ceil(Log2(refH)))) \quad (G\text{-}8)$$

3. The variables scaleX and scaleY are derived by

$$scaleX = ((refW << shiftX) + (scaledW >> 1)) / scaledW \quad (G\text{-}9)$$
$$scaleY = ((refH << shiftY) + (scaledH >> 1)) / scaledH \quad (G\text{-}10)$$

The reference layer luma location (xRef, yRef) relative to the upper-left sample of the reference layer picture is derived as specified by the following ordered steps.

1. The inverse macroblock scanning process as specified in subclause 6.4.1 is invoked with CurrMbAddr as the input and the output is assigned to (xM, yM). For this invocation of the process in subclause 6.4.1, the current macroblock is treated as field macroblock when fieldMbFlag is equal to 1, and it is treated as frame macroblock when fieldMbFlag is equal to 0.
2. The luma location (xC, yC) is derived by

$$xC = xM + xP \quad (G\text{-}11)$$
$$yC = yM + yP * (1 + fieldMbFlag - field\_pic\_flag) \quad (G\text{-}12)$$

3. The reference layer luma location is derived by

$$xRef = ((xC - offsetX) * scaleX + (1 << (shiftX - 1))) >> shiftX \quad (G\text{-}13)$$
$$yRef = ((yC - offsetY) * scaleY + (1 << (shiftY - 1))) >> shiftY \quad (G\text{-}14)$$

The reference layer macroblock address mbAddrRefLayer and a luma location (xB, yB) relative to the upper-left sample of the reference layer macroblock mbAddrRefLayer are derived as follows.

– If MbaffFrameFlag is equal to 0 and RefLayerMbaffFrameFlag is equal to 0, the following ordered steps are specified.

1. The reference layer macroblock address mbAddrRefLayer is derived as follows.
   – If any of the following conditions is true, mbAddrRefLayer is marked as not available.
     – xRef is less than 0 or xRef is greater than or equal to $RefLayerPicWidthInSamples_L$
     – yRef is less than 0 or yRef is greater than or equal to $RefLayerPicHeightInSamples_L$
   – Otherwise, the macroblock address mbAddrRefLayer is derived by

$$mbAddrRefLayer = (yRef / 16) * RefLayerPicWidthInMbs + (xRef / 16) \quad (G\text{-}15)$$

2. The luma location (xB, yB) is derived as follows.
   – If mbAddrRefLayer is not available, (xB, yB) is marked as not available.
   – Otherwise (mbAddrRefLayer is available), (xB, yB) is set equal to (xRef % 16, yRef % 16).

– Otherwise (MbaffFrameFlag is equal to 1 or RefLayerMbaffFrameFlag is equal to 1), the following ordered steps are specified.

NOTE 2 – When MbaffFrameFlag is equal to 1 or RefLayerMbaffFrameFlag is equal to 1, field_pic_flag and RefLayerFieldPicFlag are both equal to 0 (see subclause G.7.4.3.4).

1. A virtual reference layer macroblock address virtMbAddrRefLayer is derived as follows.
   – If any of the following conditions is true, virtMbAddrRefLayer is marked as not available.
     – xRef is less than 0 or xRef is greater than or equal to $RefLayerPicWidthInSamples_L$
     – yRef is less than 0 or yRef is greater than or equal to $RefLayerPicHeightInSamples_L$
   – Otherwise, the following applies.

– If RefLayerMbaffFrameFlag is equal to 1, virtMbAddrRefLayer is derived by

$$virtMbAddrRefLayer = 2 * ((yRef / 32) * RefLayerPicWidthInMbs + (xRef / 16)) + (yRef \% 32) / 16 \quad \text{(G-16)}$$

– Otherwise (RefLayerMbaffFrameFlag is equal to 0), virtMbAddrRefLayer is derived by

$$virtMbAddrRefLayer = (yRef / 16) * RefLayerPicWidthInMbs + (xRef / 16) \quad \text{(G-17)}$$

2. The reference layer macroblock address mbAddrRefLayer and the luma location (xB, yB) are derived as follows.

If virtMbAddrRefLayer is not available, the reference layer macroblock address mbAddrRefLayer and the luma location (xB, yB) are marked as not available.

– Otherwise, if fieldMbFlag is equal to 0 and refLayerFieldMbRef[virtMbAddrRefLayer] is equal to 1, the field-to-frame reference layer macroblock conversion process as specified in subclause G.6.1.1 is invoked with virtMbAddrRefLayer, (xRef, yRef), and refLayerMbType as the inputs and the outputs are assigned to mbAddrRefLayer and (xB, yB).

– Otherwise, if fieldMbFlag is equal to 1 and refLayerFieldMbRef[virtMbAddrRefLayer] is equal to 0, the frame-to-field reference layer macroblock conversion process as specified in subclause G.6.1.2 is invoked with virtMbAddrRefLayer and (xRef, yRef) as the inputs and the outputs are assigned to mbAddrRefLayer and (xB, yB).

– Otherwise (virtMbAddrRefLayer is available and fieldMbFlag is equal to refLayerFieldMbRef[virtMbAddrRefLayer]), mbAddrRefLayer and (xB, yB) are derived as follows.

$$mbAddrRefLayer = ((virtMbAddrRefLayer >> fieldMbFlag) << fieldMbFlag) + (CurrMbAddr \% 2) * fieldMbFlag \quad \text{(G-18)}$$
$$xB = (xRef \% 16) \quad \text{(G-19)}$$
$$yB = (yRef \% (16 << fieldMbFlag)) >> fieldMbFlag \quad \text{(G-20)}$$

G.6.1.1 Field-to-frame reference layer macroblock conversion process

Inputs to this process are

– a virtual reference layer macroblock address virtMbAddrRefLayer,

– a reference layer luma location (xRef, yRef) relative to the upper-left luma sample of the reference layer picture,

– a one-dimensional array refLayerMbType with RefLayerPicSizeInMbs elements specifying macroblock types for the macroblocks of the reference layer representation.

Outputs of this process are

– the macroblock address mbAddrRefLayer of the reference layer macroblock,

– a luma location (xB, yB) relative to the upper-left luma sample of the reference layer macroblock.

The macroblock addresses mbAddrRefLayerTop and mbAddrRefLayerBot are derived by

$$mbAddrRefLayerTop = virtMbAddrRefLayer - (virtMbAddrRefLayer \% 2) \quad \text{(G-21)}$$
$$mbAddrRefLayerBot = mbAddrRefLayerTop + 1 \quad \text{(G-22)}$$

The reference layer macroblock address mbAddrRefLayer is derived as follows.

– If refLayerMbType[mbAddrRefLayerTop] is equal to I_PCM, I_16x16, I_8x8, I_4x4, or I_BL, mbAddrRefLayer is set equal to mbAddrRefLayerBot.

– Otherwise (refLayerMbType[mbAddrRefLayerTop] is not equal to I_PCM, I_16x16, I_8x8, I_4x4, or I_BL), mbAddrRefLayer is set equal to mbAddrRefLayerTop.

The luma location (xB, yB) is derived by

$$xB = xRef \% 16 \quad \text{(G-23)}$$
$$yB = 8 * ((yRef / 16) \% 2) + 4 * ((yRef \% 16) / 8) \quad \text{(G-24)}$$

G.6.1.2 Frame-to-field reference layer macroblock conversion process

Inputs to this process are

– a virtual reference layer macroblock address virtMbAddrRefLayer,

– a virtual reference layer luma location (xRef, yRef) relative to the upper-left luma sample of the reference layer picture.

Outputs of this process are

– the macroblock address mbAddrRefLayer of the reference layer macroblock,

– a luma location (xB, yB) relative to the upper-left luma sample of the reference layer macroblock.

The macroblock addresses mbAddrRefLayerTop and mbAddrRefLayerBot are derived as follows.

– If RefLayerMbaffFrameFlag is equal to 1, mbAddrRefLayerTop and mbAddrRefLayerBot are derived by

$$mbAddrRefLayerTop = virtMbAddrRefLayer - (virtMbAddrRefLayer \% 2) \quad (G\text{-}25)$$
$$mbAddrRefLayerBot = mbAddrRefLayerTop + 1 \quad (G\text{-}26)$$

– Otherwise (RefLayerMbaffFrameFlag is equal to 0), mbAddrRefLayerTop and mbAddrRefLayerBot are derived by

$$mbAddrRefLayerTop = virtMbAddrRefLayer - ((yRef / 16) \% 2) * RefLayerPicWidthInMbs \quad (G\text{-}27)$$
$$mbAddrRefLayerBot = mbAddrRefLayerTop + RefLayerPicWidthInMbs \quad (G\text{-}28)$$

The reference layer macroblock address is derived as follows.

– If (yRef % 16) is less than 8, mbAddrRefLayer is set equal to mbAddrRefLayerTop.

– Otherwise ((yRef % 16) is greater than or equal to 8), mbAddrRefLayer is set equal to mbAddRefLayerBot.

The luma location (xB, yB) is derived by

$$xB = xRef \% 16 \quad (G\text{-}29)$$
$$yB = 8 * (((yRef \% 16) / 4) \% 2) + 4 * ((yRef / 16) \% 2) \quad (G\text{-}30)$$

G.6.2 Derivation process for reference layer partitions

This process is only invoked when no_inter_layer_pred_flag is equal to 0.

Inputs to this process are

– a luma location (xP, yP) relative to the upper-left luma sample of the current macroblock,

– a variable fieldMbFlag specifying whether the current macroblock is a field or a frame macroblock,

– a one-dimensional array refLayerFieldMbFlag with RefLayerPicSizeInMbs elements specifying which macroblocks of the reference layer representation are field macroblocks and which macroblocks are frame macroblocks,

– a one-dimensional array refLayerMbType with RefLayerPicSizeInMbs elements specifying macroblock types for the macroblocks of the reference layer representation,

– a (RefLayerPicSizeInMbs)x4 array refLayerSubMbType specifying the sub-macroblock types for the macroblocks of the reference layer representation.

Outputs of this process are

– the macroblock address mbAddrRefLayer specifying the reference layer macroblock,

– the macroblock partition index mbPartIdxRefLayer specifying the reference layer macroblock partition inside the reference layer macroblock mbAddrRefLayer,

– the sub-macroblock partition index subMbPartIdxRefLayer specifying the reference layer sub-macroblock partition inside the macroblock partition mbPartIdxRefLayer of the reference layer macroblock mbAddrRefLayer.

The derivation process for reference layer macroblocks as specified in subclause G.6.1 is invoked with the luma location (xP, yP), fieldMbFlag, refLayerFieldMbFlag, and refLayerMbType as the input and the outputs are assigned to mbAddrRefLayer and (xB, yB).

The reference layer macroblock partition index mbPartIdxRefLayer and the reference layer sub-macroblock partition index subMbPartIdxRefLayer are derived as follows.

- If mbAddrRefLayer is not available, mbPartIdxRefLayer and subMbPartIdxRefLayer are marked as not available.
- Otherwise, if refLayerMbType[mbAddrRefLayer] is equal to I_PCM, I_16x16, I_8x8, I_4x4, or I_BL, mbPartIdxRefLayer and subMbPartIdxRefLayer are set equal to 0.
- Otherwise, if refLayerMbType[mbAddrRefLayer] is equal to B_Skip or B_Direct_16x16, the following applies.
  - If ref_layer_dq_id is greater than 0, mbPartIdxRefLayer and subMbPartIdxRefLayer are set equal to 0.
  - Otherwise (ref_layer_dq_id is equal to 0), mbPartIdxRefLayer and subMbPartIdxRefLayer are derived by

$$mbPartIdxRefLayer = 2 * (yB / 8) + (xB / 8) \quad \text{(G-31)}$$
$$subMbPartIdxRefLayer = 2 * ((yB \% 8) / 4) + ((xB \% 8) / 4) \quad \text{(G-32)}$$

- Otherwise (mbAddrRefLayer is available and refLayerMbType[mbAddrRefLayer] is not equal to I_PCM, I_16x16, I_8x8, I_4x4, I_BL, B_Skip, or B_Direct_16x16), the following ordered steps are specified.
  1. The reference layer macroblock partition index mbPartIdxRefLayer is set equal to the macroblock partition index of the macroblock partition inside the reference layer macroblock mbAddrRefLayer that covers the luma location (xB, yB). The macroblock partitioning of the reference layer macroblock mbAddrRefLayer is given by the macroblock type refLayerMbType[mbAddrRefLayer] and the specification in subclause G.8.4.1.
  2. The reference layer sub-macroblock partition index subMbPartIdxRefLayer is derived as follows.
     - If NumMbPart(refLayerMbType[mbAddrRefLayer]) is less than 4, subMbPartIdxRefLayer is set equal to 0.
     - Otherwise (NumMbPart(refLayerMbType[mbAddrRefLayer]) is equal to 4), the following applies.
       - If refLayerSubMbType[mbAddrRefLayer][mbPartIdxRefLayer] is equal to B_Direct_8x8, the following applies.
         - If ref_layer_dq_id is greater than 0, subMbPartIdxRefLayer is set equal to 0.
         - Otherwise (ref_layer_dq_id is equal to 0), subMbPartIdxRefLayer is derived by

$$subMbPartIdxRefLayer = 2 * ((yB \% 8) / 4) + ((xB \% 8) / 4) \quad \text{(G-33)}$$

       - Otherwise (refLayerSubMbType[mbAddrRefLayer][mbPartIdxRefLayer] is not equal to B_Direct_8x8), the reference layer sub-macroblock partition index subMbPartIdxRefLayer is set equal to the sub-macroblock partition index of the sub-macroblock partition inside the macroblock partition mbPartIdxRefLayer of the reference layer macroblock mbAddrRefLayer that covers the luma location (xB, yB). The sub-macroblock partitioning of the reference layer macroblock partition mbPartIdxRefLayer inside the reference layer macroblock mbAddrRefLayer is given by the sub-macroblock type refLayerSubMbType[mbAddrRefLayer][mbPartIdxRefLayer] and the specification in subclause G.8.4.1.

G.6.3 Derivation process for reference layer sample locations in resampling

Inputs to this process are

- a variable chromaFlag specifying whether the luma or a chroma component is subject to the resampling process,
- a sample location (xP, yP) relative to the upper-left sample of the current macroblock,
- a variable fieldMbFlag specifying whether the current macroblock is a field or a frame macroblock,
- a variable botFieldFlag specifying whether a top or a bottom field is subject to the resampling process (when RefLayerFrameMbsOnlyFlag is equal to 0 or frame_mbs_only_flag is equal to 0).

Output of this process is a reference layer sample location (xRef16, yRef16), which specifies the following.

– If RefLayerFrameMbsOnlyFlag is equal to 1 or RefLayerFieldPicFlag is equal to 1, (xRef16, yRef16) specifies the reference layer sample location in units of 1/16-th sample relative to the upper-left sample of the reference layer picture.

– Otherwise (RefLayerFrameMbsOnlyFlag is equal to 0 and RefLayerFieldPicFlag is equal to 0), (xRef16, yRef16) specifies the reference layer sample location in units of 1/16-th field sample relative to the upper-left sample of the field specified by botFieldFlag of the reference layer picture.

The variables subW, subH, shiftX, shiftY, scaleX, scaleY, offsetX, offsetY, addX, addY, deltaX, and deltaY are derived as specified in the following ordered steps.

NOTE – These variables do not depend on the input sample location (xP, yP), the input variable fieldMbFlag, or the current macroblock address CurrMbAddr.

1. With Z being replaced by L for chromaFlag equal to 0 and C for chromaFlag equal to 1, the variables refW, refH, scaledW, and scaledH are derived by

$$refW = RefLayerPicWidthInSamples_Z \quad (G\text{-}34)$$
$$refH = RefLayerPicHeightInSamples_Z * (1 + RefLayerFieldPicFlag) \quad (G\text{-}35)$$
$$scaledW = ScaledRefLayerPicWidthInSamples_Z \quad (G\text{-}36)$$
$$scaledH = ScaledRefLayerPicHeightInSamples_Z * (1 + field\_pic\_flag) \quad (G\text{-}37)$$

2. When frame_mbs_only_flag is equal to 0 and RefLayerFrameMbsOnlyFlag is equal to 1, the variable scaledH is modified by

$$scaledH = scaledH / 2 \quad (G\text{-}38)$$

3. The variables refPhaseX, refPhaseY, phaseX, phaseY, subW, and subH are derived by

$$refPhaseX = ((chromaFlag == 0) ? 0 : (ref\_layer\_chroma\_phase\_x\_plus1\_flag - 1)) \quad (G\text{-}39)$$
$$refPhaseY = ((chromaFlag == 0) ? 0 : (ref\_layer\_chroma\_phase\_y\_plus1 - 1)) \quad (G\text{-}40)$$
$$phaseX = ((chromaFlag == 0) ? 0 : (chroma\_phase\_x\_plus1\_flag - 1)) \quad (G\text{-}41)$$
$$phaseY = ((chromaFlag == 0) ? 0 : (chroma\_phase\_y\_plus1 - 1)) \quad (G\text{-}42)$$
$$subW = ((chromaFlag == 0) ? 1 : SubWidthC) \quad (G\text{-}43)$$
$$subH = ((chromaFlag == 0) ? 1 : SubHeightC) \quad (G\text{-}44)$$

4. When RefLayerFrameMbsOnlyFlag is equal to 0 or frame_mbs_only_flag is equal to 0, the following applies.

   – The variable phaseY is modified by

$$phaseY = phaseY + 4 * botFieldFlag \quad (G\text{-}45)$$

   – The variable refPhaseY is modified as follows.

     – If RefLayerFrameMbsOnlyFlag is equal to 1, the variable refPhaseY is modified by

$$refPhaseY = 2 * refPhaseY + 2 \quad (G\text{-}46)$$

     – Otherwise (RefLayerFrameMbsOnlyFlag is equal to 0), the variable refPhaseY is modified by

$$refPhaseY = refPhaseY + 4 * botFieldFlag \quad (G\text{-}47)$$

5. The variables shiftX and shiftY are derived by

$$shiftX = ((level\_idc <= 30) ? 16 : (31 - Ceil(Log2(refW)))) \quad (G\text{-}48)$$
$$shiftY = ((level\_idc <= 30) ? 16 : (31 - Ceil(Log2(refH)))) \quad (G\text{-}49)$$

6. The variables scaleX and scaleY are derived by

$$scaleX = ((refW << shiftX) + (scaledW >> 1)) / scaledW \quad (G\text{-}50)$$
$$scaleY = ((refH << shiftY) + (scaledH >> 1)) / scaledH \quad (G\text{-}51)$$

7. The variables offsetX, addX, and deltaX are derived by

$$offsetX = ScaledRefLayerLeftOffset / subW \quad (G\text{-}52)$$

$$addX = (((refW * (2 + phaseX)) << (shiftX - 2)) + (scaledW >> 1)) / scaledW + (1 << (shiftX - 5)) \quad (G\text{-}53)$$

$$deltaX = 4 * (2 + refPhaseX) \quad (G\text{-}54)$$

8. The variables offsetY, addY, and deltaY are derived as follows.

   – If RefLayerFrameMbsOnlyFlag is equal to 1 and frame_mbs_only_flag is equal to 1, the variables offsetY, addY, and deltaY are derived by

$$offsetY = ScaledRefLayerTopOffset / subH \quad (G\text{-}55)$$

$$addY = (((refH * (2 + phaseY)) << (shiftY - 2)) + (scaledH >> 1)) / scaledH + (1 << (shiftY - 5)) \quad (G\text{-}56)$$

$$deltaY = 4 * (2 + refPhaseY) \quad (G\text{-}57)$$

   – Otherwise (RefLayerFrameMbsOnlyFlag is equal to 0 or frame_mbs_only_flag is equal to 0), the variables offsetY, addY, and deltaY are derived by

$$offsetY = ScaledRefLayerTopOffset / (2 * subH) \quad (G\text{-}58)$$

$$addY = (((refH * (2 + phaseY)) << (shiftY - 3)) + (scaledH >> 1)) / scaledH + (1 << (shiftY - 5)) \quad (G\text{-}59)$$

$$deltaY = 2 * (2 + refPhaseY) \quad (G\text{-}60)$$

The sample location (xC, yC) is derived as specified in the following ordered steps.

1. The inverse macroblock scanning process as specified in subclause 6.4.1 is invoked with CurrMbAddr as input and the output is assigned to (xM, yM). For this invocation of the process in subclause 6.4.1, the current macroblock is treated as field macroblock when fieldMbFlag is equal to 1 and it is treated as frame macroblock when fieldMbFlag is equal to 0.

2. The sample location (xC, yC) is derived by

$$xC = xP + (xM >> (subW - 1)) \quad (G\text{-}61)$$

$$yC = yP + (yM >> (subH - 1 + fieldMbFlag - field\_pic\_flag)) \quad (G\text{-}62)$$

3. When RefLayerFrameMbsOnlyFlag is equal to 0 or frame_mbs_only_flag is equal to 0, the vertical component of the sample location (xC, yC) is modified by

$$yC = yC >> (1 - fieldMbFlag) \quad (G\text{-}63)$$

The reference layer sample location (xRef16 yRef16) is derived by

$$xRef16 = (((xC - offsetX) * scaleX + addX) >> (shiftX - 4)) - deltaX \quad (G\text{-}64)$$

$$yRef16 = (((yC - offsetY) * scaleY + addY) >> (shiftY - 4)) - deltaY \quad (G\text{-}65)$$

G.7 Syntax and semantics

This clause specifies syntax and semantics for coded video sequences that conform to one or more of the profiles specified in this Annex.

G.7.1 Method of specifying syntax in tabular form

The specifications in subclause 7.1 apply.

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

The specifications in subclauses 7.2 apply. Additionally, the following syntax function is specified.

in_crop_window(mbAddr) is specified by the following ordered steps.

1. The variables mbX is set equal to (mbAddr % PicWidthInMbs).

2. The variables mbY0 and mbY1 are derived as follows.
   - If MbaffFrameFlag is equal to 0, mbY0 and mbY1 are set equal to (mbAddr / PicWidthInMbs).
   - Otherwise (MbaffFrameFlag is equal to 1), mbY0 is set equal to (2 * ((mbAddr / PicWidthInMbs) / 2)) and mbIdxY1 is set equal to (mbIdxY0 + 1).
3. The variable scalMbH is set equal to (16 * (1 + field_pic_flag)).
4. The return value of in_crop_window (mbAddr) is derived as follows.
   - If all of the following conditions are true, the return value of in_crop_window(mbAddr) is equal to TRUE.
     - no_inter_layer_pred_flag is equal to 0
     - mbX is greater than or equal to ((ScaledRefLayerLeftOffset + 15) / 16)
     - mbX is less than ((ScaledRefLayerLeftOffset + ScaledRefLayerPicWidthInSamples$_L$) / 16)
     - mbY0 is greater than or equal to ((ScaledRefLayerTopOffset + scalMbH – 1) / scalMbH)
     - mbY1 is less than ((ScaledRefLayerTopOffset + ScaledRefLayerPicHeightInSamples$_L$) / scalMbH)
   - Otherwise, the return value of in_crop_window(mbAddr) is equal to FALSE.

G.7.3 Syntax in tabular form

G.7.3.1 NAL unit syntax

The syntax table is specified in subclause 7.3.1.

G.7.3.1.1 NAL unit header SVC extension syntax

nal_unit_header_svc_extension( ) {	C	Descriptor
reserved_one_bit	All	u(1)
idr_flag	All	u(1)
priority_id	All	u(6)
no_inter_layer_pred_flag	All	u(1)
dependency_id	All	u(3)
quality_id	All	u(4)
temporal_id	All	u(3)
use_ref_base_pic_flag	All	u(1)
discardable_flag	All	u(1)
output_flag	All	u(1)
reserved_three_2bits	All	u(2)
}		

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

G.7.3.2.1 Sequence parameter set RBSP syntax

The syntax table is specified in subclause 7.3.2.1.

G.7.3.2.1.1 Sequence parameter set data syntax

The syntax table is specified in subclause 7.3.2.1.1.

G.7.3.2.1.1.1 Scaling list syntax

The syntax table is specified in subclause 7.3.2.1.1.1.

G.7.3.2.1.2 Sequence parameter set extension RBSP syntax

The syntax table is specified in subclause 7.3.2.1.2.

G.7.3.2.1.3 Subset sequence parameter set RBSP syntax

subset_seq_parameter_set_rbsp( ) {	C	Descriptor
seq_parameter_set_data( )	0	
if(profile_idc == 83 || profile_idc == 86) {		
seq_parameter_set_svc_extension( )	0	
svc_vui_parameters_present_flag	0	u(1)
if(svc_vui_parameters_present_flag == 1)		
svc_vui_parameters_extension( )	0	
}		
additional_extension2_flag	0	u(1)
if(additional_extension2_flag == 1)		
while(more_rbsp_data( ))		
additional_extension2_data_flag	0	u(1)
rbsp_trailing_bits( )	0	
}		

G.7.3.2.1.4 Sequence parameter set SVC extension syntax

seq_parameter_set_svc_extension( ) {	C	Descriptor
inter_layer_deblocking_filter_control_present_flag	0	u(1)
extended_spatial_scalability	0	u(2)
if(ChromaArrayType == 1 || ChromaArrayType == 2)		
chroma_phase_x_plus1_flag	0	u(1)
if(ChromaArrayType == 1)		
chroma_phase_y_plus1	0	u(2)
if(extended_spatial_scalability == 1) {		
if(ChromaArrayType > 0) {		
seq_ref_layer_chroma_phase_x_plus1_flag	0	u(1)
seq_ref_layer_chroma_phase_y_plus1	0	u(2)
}		
seq_scaled_ref_layer_left_offset	0	se(v)
seq_scaled_ref_layer_top_offset	0	se(v)
seq_scaled_ref_layer_right_offset	0	se(v)
seq_scaled_ref_layer_bottom_offset	0	se(v)
}		
seq_tcoeff_level_prediction_flag	0	u(1)
if(seq_tcoeff_level_prediction_flag) {		
adaptive_tcoeff_level_prediction_flag	0	u(1)
}		
slice_header_restriction_flag	0	u(1)
}		

G.7.3.2.2 Picture parameter set RBSP syntax

The syntax table is specified in subclause 7.3.2.2.

G.7.3.2.3 Supplemental enhancement information RBSP syntax

The syntax table is specified in subclause 7.3.2.3.

G.7.3.2.3.1 Supplemental enhancement information message syntax

The syntax table is specified in subclause 7.3.2.3.1.

G.7.3.2.4 Access unit delimiter RBSP syntax

The syntax table is specified in subclause 7.3.2.4.

G.7.3.2.5 End of sequence RBSP syntax

The syntax table is specified in subclause 7.3.2.5.

G.7.3.2.6 End of stream RBSP syntax

The syntax table is specified in subclause 7.3.2.6.

G.7.3.2.7 Filler data RBSP syntax

The syntax table is specified in subclause 7.3.2.7.

G.7.3.2.8 Slice layer without partitioning RBSP syntax

The syntax table is specified in subclause 7.3.2.8.

G.7.3.2.9 Slice data partition RBSP syntax

Slice data partition syntax is not present in coded video sequences conforming to one or more of the profiles specified in this Annex.

G.7.3.2.10 RBSP slice trailing bits syntax

The syntax table is specified in subclause 7.3.2.10.

G.7.3.2.11 RBSP trailing bits syntax

The syntax table is specified in subclause 7.3.2.11.

G.7.3.2.12 Prefix NAL unit RBSP syntax

prefix_nal_unit_rbsp( ) {	C	Descriptor
if(nal_ref_idc != 0) {		
store_ref_base_pic_flag	2	u(1)
if((use_ref_base_pic_flag || store_ref_base_pic_flag) && !idr_flag)		
dec_ref_base_pic_marking( )	2	
prefix_nal_unit_additional_extension_flag	2	u(1)
if(prefix_nal_unit_additional_extension_flag == 1)		
while(more_rbsp_data( ))		
prefix_nal_unit_extension_flag	2	u(1)
rbsp_trailing_bits( )	2	
}		
}		

G.7.3.2.13 Slice layer in scalable extension RBSP syntax

slice_layer_in_scalable_extension_rbsp( ) {	C	Descriptor
slice_header_in_scalable_extension( )	2	
if(!slice_skip_flag)		
slice_data_in_scalable_extension( )	2 | 3 | 4	
rbsp_slice_trailing_bits( )	2	
}		

G.7.3.3 Slice header syntax

The syntax table is specified in subclause 7.3.3.

G.7.3.3.1 Reference picture list reordering syntax

The syntax table is specified in subclause 7.3.3.1.

G.7.3.3.2 Prediction weight table syntax

The syntax table is specified in subclause 7.3.3.2.

G.7.3.3.3 Decoded reference picture marking syntax

The syntax table is specified in subclause 7.3.3.3.

G.7.3.3.4 Slice header in scalable extension syntax

slice_header_in_scalable_extension( ) {	C	Descriptor
first_mb_in_slice	2	ue(v)
slice_type	2	ue(v)
pic_parameter_set_id	2	ue(v)
if(separate_colour_plane_flag == 1)		
colour_plane_id	2	u(2)
frame_num	2	u(v)
if(!frame_mbs_only_flag) {		
field_pic_flag	2	u(1)
if(field_pic_flag)		
bottom_field_flag	2	u(1)
}		
if(idr_flag == 1)		
idr_pic_id	2	ue(v)
if(pic_order_cnt_type == 0) {		
pic_order_cnt_lsb	2	u(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt_bottom	2	se(v)
}		
if(pic_order_cnt_type == 1 && !delta_pic_order_always_zero_flag) {		
delta_pic_order_cnt[0 **]**	2	se(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt[1 **]**	2	se(v)
}		
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	2	ue(v)

if(quality_id == 0) {		
if(slice_type == EB)		
direct_spatial_mv_pred_flag	2	u(1)
if(slice_type == EP || slice_type == EB) {		
num_ref_idx_active_override_flag	2	u(1)
if(num_ref_idx_active_override_flag) {		
num_ref_idx_l0_active_minus1	2	ue(v)
if(slice_type == EB)		
num_ref_idx_l1_active_minus1	2	ue(v)
}		
}		
ref_pic_list_reordering( )	2	
if((weighted_pred_flag && slice_type == EP) || (weighted_bipred_idc == 1 && slice_type == EB)) {		
if(!no_inter_layer_pred_flag)		
base_pred_weight_table_flag	2	u(1)
if(no_inter_layer_pred_flag || !base_pred_weight_table_flag)		
pred_weight_table( )	2	
}		
if(nal_ref_idc != 0) {		
dec_ref_pic_marking( )	2	
if(!slice_header_restriction_flag) {		
store_ref_base_pic_flag	2	u(1)
if((use_ref_base_pic_flag || store_ref_base_pic_flag) && !idr_flag)		
dec_ref_base_pic_marking( )	2	
}		
}		
}		
if(entropy_coding_mode_flag && slice_type != EI)		
cabac_init_idc	2	ue(v)
slice_qp_delta	2	se(v)
if(deblocking_filter_control_present_flag) {		
disable_deblocking_filter_idc	2	ue(v)
if(disable_deblocking_filter_idc != 1) {		
slice_alpha_c0_offset_div2	2	se(v)
slice_beta_offset_div2	2	se(v)
}		
}		
if(quality_id == 0 && num_slice_groups_minus1 > 0 && slice_group_map_type >= 3 && slice_group_map_type <= 5)		
slice_group_change_cycle	2	u(v)
if(!no_inter_layer_pred_flag && quality_id == 0) {		
ref_layer_dq_id	2	ue(v)
if(inter_layer_deblocking_filter_control_present_flag) {		
disable_inter_layer_deblocking_filter_idc	2	ue(v)
if(disable_inter_layer_deblocking_filter_idc != 1) {		
inter_layer_slice_alpha_c0_offset_div2	2	se(v)
inter_layer_slice_beta_offset_div2	2	se(v)

}		
}		
constrained_intra_resampling_flag	2	u(1)
if(extended_spatial_scalability == 2) {		
if(ChromaArrayType > 0) {		
ref_layer_chroma_phase_x_plus1_flag	2	u(1)
ref_layer_chroma_phase_y_plus1	2	u(2)
}		
scaled_ref_layer_left_offset	2	se(v)
scaled_ref_layer_top_offset	2	se(v)
scaled_ref_layer_right_offset	2	se(v)
scaled_ref_layer_bottom_offset	2	se(v)
}		
}		
if(!no_inter_layer_pred_flag) {		
slice_skip_flag	2	u(1)
if(slice_skip_flag)		
num_mbs_in_slice_minus1	2	ue(v)
else {		
adaptive_base_mode_flag	2	u(1)
if(!adaptive_base_mode_flag)		
default_base_mode_flag	2	u(1)
if(!default_base_mode_flag) {		
adaptive_motion_prediction_flag	2	u(1)
if(!adaptive_motion_prediction_flag)		
default_motion_prediction_flag	2	u(1)
}		
adaptive_residual_prediction_flag	2	u(1)
if(adaptive_residual_prediction_flag)		
default_residual_prediction_flag	2	u(1)
}		
if(adaptive_tcoeff_level_prediction_flag == 1)		
tcoeff_level_prediction_flag	2	u(1)
}		
if(!slice_header_restriction_flag) {		
scan_idx_start	2	u(4)
scan_idx_end	2	u(4)
}		
}		

G.7.3.3.5 Decoded reference base picture marking syntax

dec_ref_base_pic_marking( ) {	C	Descriptor
adaptive_ref_base_pic_marking_mode_flag	2 | 5	u(1)
if(adaptive_ref_base_pic_marking_mode_flag)		
do {		
memory_management_control_operation	2 | 5	ue(v)
if(memory_management_control_operation == 1)		
difference_of_pic_nums_minus1	2 | 5	ue(v)
if(memory_management_control_operation == 2)		
long_term_pic_num	2 | 5	ue(v)
} while(memory_management_control_operation != 0)		
}		

G.7.3.4 Slice data syntax

The syntax table is specified in subclause 7.3.4.

G.7.3.4.1 Slice data in scalable extension syntax

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

prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && (CurrMbAddr % 2) == 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

G.7.3.5 Macroblock layer syntax

The syntax table is specified in subclause 7.3.5.

G.7.3.5.1 Macroblock prediction syntax

The syntax table is specified in subclause 7.3.5.1.

G.7.3.5.2 Sub-macroblock prediction syntax

The syntax table is specified in subclause 7.3.5.2.

G.7.3.5.3 Residual data syntax

The syntax table is specified in subclause 7.3.5.3.

G.7.3.5.3.1 Residual luma syntax

The syntax table is specified in subclause 7.3.5.3.1.

G.7.3.5.3.2 Residual block CAVLC syntax

The syntax table is specified in subclause 7.3.5.3.2.

G.7.3.5.3.3 Residual block CABAC syntax

The syntax table is specified in subclause 7.3.5.3.3.

G.7.3.6 Macroblock layer in scalable extension syntax

macroblock_layer_in_scalable_extension( ) {	C	Descriptor
if(in_crop_window(CurrMbAddr) && adaptive_base_mode_flag)		
base_mode_flag	2	u(1) | ae(v)
if(!base_mode_flag)		
mb_type	2	ue(v) | ae(v)
if(!base_mode_flag && mb_type == I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i]	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i]	3	u(v)
} else {		
if(! base_mode_flag) {		
noSubMbPartSizeLessThan8x8Flag = 1		

if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) == 4) {		
sub_mb_pred_in_scalable_extension(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type [mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type == I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
mb_pred_in_scalable_extension(mb_type)	2	
}		
}		
if(adaptive_residual_prediction_flag && slice_type != EI && (base_mode_flag || (MbPartPredMode(mb_type, 0) != Intra_16x16 && MbPartPredMode(mb_type, 0) != Intra_8x8 && MbPartPredMode(mb_type, 0) != Intra_4x4 && mb_type != I_PCM && in_crop_window(CurrMbAddr))))		
residual_prediction_flag	2	u(1) | ae(v)
if(scan_idx_end >= scan_idx_start) {		
if(base_mode_flag || MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && (base_mode_flag || (mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		
transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || (!base_mode_flag && MbPartPredMode(mb_type, 0) == Intra_16x16)) {		
mb_qp_delta	2	se(v) | ae(v)
residual(base_mode_flag, scan_idx_start, scan_idx_end)	3 | 4	
}		
}		
}		
}		

G.7.3.6.1 Macroblock prediction in scalable extension syntax

mb_pred_in_scalable_extension(mb_type) {	C	Descriptor
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_8x8 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx = 0; luma4x4BlkIdx < 16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag[luma4x4BlkIdx])		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
if(MbPartPredMode(mb_type, 0) == Intra_8x8)		
for(luma8x8BlkIdx = 0; luma8x8BlkIdx < 4; luma8x8BlkIdx++) {		
prev_intra8x8_pred_mode_flag[luma8x8BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra8x8_pred_mode_flag[luma8x8BlkIdx])		
rem_intra8x8_pred_mode[luma8x8BlkIdx **]**	2	u(3) | ae(v)
}		
if(ChromaArrayType != 0)		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
if(in_crop_window(CurrMbAddr) && adaptive_motion_prediction_flag) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
motion_prediction_flag_l0[mbPartIdx **]**	2	u(1) | ac(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
motion_prediction_flag_l1[mbPartIdx **]**	2	u(1) | ae(v)
}		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1 && !motion_prediction_flag_l0[mbPartIdx])		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0 && !motion_prediction_flag_l1[mbPartIdx])		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

G.7.3.6.2 Sub-macroblock prediction in scalable extension syntax

sub_mb_pred_in_scalable_extension(mb_type) {	C	Descriptor
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
if(in_crop_window(CurrMbAddr) && adaptive_motion_prediction_flag) {		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(SubMbPredMode(sub_mb_type[mbPartIdx]) != Direct && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
motion_prediction_flag_l0[mbPartIdx]	2	u(1) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(SubMbPredMode(sub_mb_type[mbPartIdx]) != Direct && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
motion_prediction_flag_l1[mbPartIdx]	2	u(1) | ae(v)
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1 && !motion_prediction_flag_l0[mbPartIdx])		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0 && !motion_prediction_flag_l1[mbPartIdx])		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

G.7.4 Semantics

Semantics associated with the syntax structures and syntax elements within these structures (in subclause G.7.3 and in subclause 7.3 by reference in G.7.3) are specified in this subclause and by reference to subclause 7.4. When the semantics of a syntax element are specified using a table or a set of tables, any values that are not specified in the table(s) shall not be present in the bitstream unless otherwise specified in this Recommendation | International Standard.

One or more sub-bitstreams may be derived from a bitstream conforming to one or more of the profiles specified in this annex. Sub-bitstreams that are derived according to the process specified in subclause G.8.8.1 conform to one or more of the profiles specified in Annex A or one or more of the profiles specified in this annex.

One or more sub-bitstreams conform to the profiles specified in Annex A. The decoding for these sub-bitstreams is specified in clauses 2-9 and Annexes B-E.

The decoding for bitstreams conforming to one or more of the profiles specified in this Annex is completely specified in this Annex with reference made to clauses 2-9 and Annexes B-E.

A specification or a process in clauses 2-9 and Annexes B-E may be used as is or by specifying assignments or alternative meanings of certain parts.

G.7.4.1 NAL unit semantics

The semantics for the syntax elements in subclause G.7.3.1 are specified in subclause 7.4.1. The following specifications additionally apply.

For NAL units with nal_unit_type equal to 14, nal_ref_idc shall be identical to nal_ref_idc of the associated NAL unit, which succeeds the NAL unit with nal_unit_type equal to 14 in decoding order.

The value of nal_ref_idc shall be identical for all VCL NAL units of a dependency representation.

The variable refNalRefIdc is derived as follows.

– When nal_unit_type is not equal to 20 or dependency_id is equal to the minimum value of dependency_id for all VCL NAL units of the access unit, refNalRefIdc is set equal to 0.

– Otherwise, refNalRefIdc is set equal to the maximum value of nal_ref_idc for all VCL NAL units of the access unit with a value of dependency_id less than the current value of dependency_id.

When refNalRefIdc is greater than 0, the value of nal_ref_idc shall not be equal to 0.

nal_ref_idc equal to 0 for a NAL unit containing a slice and having a value of dependency_id that is equal to the maximum value of dependency_id in the access unit indicates that all coded slice NAL units of the access unit are coded slice NAL units of a non-reference picture.

nal_ref_idc greater than 0 for a NAL unit containing a slice and having a value of dependency_id that is equal to the maximum value of dependency_id in the access unit indicates that all coded slice NAL units of the access unit are coded slice NAL units of a non-reference picture.

G.7.4.1.1 NAL unit header SVC extension semantics

The syntax elements idr_flag, priority_id, no_inter_layer_pred_flag, dependency_id, quality_id, temporal_id, use_ref_base_pic_flag, discardable_flag, and output_flag, when present in a prefix NAL unit, are considered to apply to the associated NAL unit.

reserved_one_bit shall be equal to 1. The value of reserved_one_bit may be specified by future extension of this Recommendation | International Standard. Decoders shall ignore the value of reserved_one_bit.

idr_flag equal to 1 specifies that the current access unit is an IDR access unit when the value of dependency_id for the NAL unit is equal to the maximum value of dependency_id for the access unit.

The value of idr_flag shall be the same for all NAL units of a dependency representation. For filler prefix NAL units, idr_flag shall be equal to 0. When idr_flag is equal to 1 for a VCL prefix NAL unit, the associated NAL unit shall have nal_unit_type equal to 5. When idr_flag is equal to 0 for a VCL prefix NAL unit, the associated NAL unit shall have nal_unit_type equal to 1.

When nal_ref_idc is equal to 0, the value of idr_flag shall be equal to 0.

For NAL units, in which idr_flag is present, the variable IdrPicFlag derived in subclause 7.4.1 is modified by setting it equal to idr_flag.

Let dIdMax be the maximum value of dependency_id in the NAL units of an access unit. An access unit is referred to as an IDR access unit when IdrPicFlag is equal to 1 for any NAL unit with dependency_id equal to dIdMax.

NOTE 1 – The classification of an access unit as IDR access unit and the partitioning of a sequence of access units in coded video sequences depends on the maximum present value of dependency_id in the associated NAL units. When NAL units are removed from a bitstream, e.g. in order to adjust the bitstream to the capabilities of a receiving device, the classification of access units as IDR access units and the partitioning of the sequence of access units into coded video sequences may change.

priority_id specifies a priority identifier for the NAL unit.

NOTE 2 – The syntax element priority_id is not required by the decoding process specified in this Recommendation | International Standard. The syntax element priority_id may be used as determined by the application within the specified constraints.

no_inter_layer_pred_flag specifies, when present in a coded slice NAL unit, whether inter-layer prediction may be used for decoding the coded slice. When no_inter_layer_pred_flag is equal to 1, inter-layer prediction is not used for

decoding the coded slice. When no_inter_layer_pred_flag is equal to 0, inter-layer prediction may or may not be used for decoding the coded slice as signalled in the macroblock layer.

For prefix NAL units, no_inter_layer_pred_flag shall be equal to 1. When nal_unit_type is equal to 20 and quality_id is greater than 0, no_inter_layer_pred_flag shall be equal to 0.

dependency_id specifies a dependency identifier for the NAL unit. dependency_id shall be equal to 0 in VCL prefix NAL units.

quality_id specifies a quality identifier for the NAL unit. quality_id shall be equal to 0 in VCL prefix NAL units.

The variable DQId is derived by

$$DQId = (dependency\_id << 4) + quality\_id \quad \text{(G-66)}$$

temporal_id specifies a temporal identifier for the NAL unit.

The value of temporal_id shall be the same for all prefix and coded slice in scalable extension NAL units of an access unit. When an access unit contains any NAL unit with nal_unit_type equal to 5 or idr_flag equal to 1, temporal_id shall be equal to 0.

The bitstream that would be obtained by invoking the bitstream extraction process as specified in subclause G.8.8.1 with pIdTarget equal to any value in the range of 0 to 63, inclusive, tIdTarget equal to any value in the range of 0 to 7, inclusive, dIdTarget equal to any value in the range of 0 to 7, inclusive, and qIdTarget equal to any value in the range of 0 to 15, inclusive, shall be conforming to this Recommendation | International Standard.

NOTE 3 – A conforming bitstream contains one or more coded slice NAL units with priority_id equal to 0, dependency_id equal to 0, quality_id equal to 0, and temporal_id equal to 0.

use_ref_base_pic_flag equal to 1 specifies that reference base pictures are used as reference pictures for the inter prediction process. use_ref_base_pic_flag equal to 0 specifies that decoded pictures are used as reference pictures during the inter prediction process.

The values of use_ref_base_pic_flag shall be the same for all NAL units of a dependency representation. use_ref_base_pic_flag shall be equal to 0 in filler prefix NAL units.

discardable_flag equal to 1 specifies that the current NAL unit is not used for decoding NAL units of the current access unit and all subsequent access units that have a greater value of dependency_id than the current NAL unit. discardable_flag equal to 0 specifies that the current NAL unit may be used for decoding NAL units of the current access unit and all subsequent access units that have a greater value of dependency_id than the current NAL unit.

discardable_flag shall be equal to 1 in filler prefix NAL units.

output_flag affects the decoded picture output and removal processes as specified in Annex C. The value of output_flag shall be the same for all NAL units of a dependency representation. The value of output_flag shall be the same for both fields of a complementary field pair.

output_flag shall be equal to 0 in filler prefix NAL units.

reserved_three_2bits shall be equal to 3. The values of reserved_three_2bits may be specified by future extension of this Recommendation | International Standard. Decoders shall ignore the value of reserved_three_2bits.

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

This subclause specifies constraints on the order of NAL units in the bitstream. Any order of NAL units in the bitstream obeying these constraints is referred to in the text as the decoding order of NAL units. Within a NAL unit, the syntax in subclauses 7.3, D.1, E.1, G.7.3, G.13.1, and G.14.1 specifies the decoding order of syntax elements. Decoders conforming to this Recommendation | International Standard shall be capable of receiving NAL units and their syntax elements in decoding order.

G.7.4.1.2.1 Order of SVC sequence parameter set RBSPs and picture parameter set RBSPs and their activation

NOTE 1 – The sequence and picture parameter set mechanism decouples the transmission of infrequently changing information from the transmission of coded macroblock data. Sequence and picture parameter sets may, in some applications, be conveyed "out-of-band" using a reliable transport mechanism.

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

Each picture parameter set RBSP is initially considered not active at the start of the operation of the decoding process. At most one picture parameter set RBSP is considered as the active picture parameter set RBSP at any given moment during the operation of the decoding process, and when any particular picture parameter set RBSP becomes the active picture parameter set RBSP, the previously-active picture parameter set RBSP (if any) is deactivated.

In addition to the active picture parameter set RBSP, zero or more picture parameter set RBSPs may be specifically active for layer representations (with a particular value of DQId less than DQIdMax) that are referred to through inter-layer prediction in decoding the target layer representation. Such a picture parameter set RBSP is referred to as active layer picture parameter set RBSP for the particular value of DQId (less than DQIdMax). The restrictions on active picture parameter set RBSPs also apply to active layer picture parameter set RBSPs with a particular value of DQId.

When a picture parameter set RBSP (with a particular value of pic_parameter_set_id) is not the active picture parameter set RBSP and it is referred to by a coded slice NAL unit with DQId equal to DQIdMax (using that value of pic_parameter_set_id), it is activated. This picture parameter set RBSP is called the active picture parameter set RBSP until it is deactivated when another picture parameter set RBSP becomes the active picture parameter set RBSP. A picture parameter set RBSP, with that particular value of pic_parameter_set_id, shall be available to the decoding process prior to its activation.

When a picture parameter set RBSP (with a particular value of pic_parameter_set_id) is not the active layer picture parameter set for a particular value of DQId less than DQIdMax and it is referred to by a coded slice NAL unit with the particular value of DQId (using that value of pic_parameter_set_id), it is activated for layer representations with the particular value of DQId. This picture parameter set RBSP is called the active layer picture parameter set RBSP for the particular value of DQId until it is deactivated when another picture parameter set RBSP becomes the active layer picture parameter set RBSP for the particular value of DQId or when decoding an access unit with DQIdMax less than or equal to the particular value of DQId. A picture parameter set RBSP, with that particular value of pic_parameter_set_id, shall be available to the decoding process prior to its activation.

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

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

Each SVC sequence parameter set RBSP is initially considered not active at the start of the operation of the decoding process. At most one SVC sequence parameter set RBSP is considered as the active SVC sequence parameter set RBSP at any given moment during the operation of the decoding process, and when any particular SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP, the previously-active SVC sequence parameter set RBSP (if any) is deactivated.

In addition to the active SVC sequence parameter set RBSP, zero or more SVC sequence parameter set RBSPs may be specifically active for layer representations (with a particular value of DQId less than DQIdMax) that are referred to through inter-layer prediction in decoding the target layer representation. Such an SVC sequence parameter set RBSP is referred to as active layer SVC sequence parameter set RBSP for the particular value of DQId (less than DQIdMax). The restrictions on active SVC sequence parameter set RBSPs also apply to active layer SVC sequence parameter set RBSPs with a particular value of DQId.

When a sequence parameter set RBSP (nal_unit_type is equal to 7) with a particular value of seq_parameter_set_id is not already the active SVC sequence parameter set RBSP and it is referred to by activation of a picture parameter set RBSP (using that value of seq_parameter_set_id) and the picture parameter set RBSP is activated by a coded slice NAL unit with DQId equal to DQIdMax (the picture parameter set RBSP becomes the active picture parameter set RBSP) and nal_unit_type equal to 1 or 5, it is activated. This sequence parameter set RBSP, is called the active SVC sequence parameter set RBSP until it is deactivated when another SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP. A sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

When a sequence parameter set RBSP (nal_unit_type is equal to 7) with a particular value of seq_parameter_set_id is not already the active SVC sequence parameter set RBSP and it is referred to by an SEI NAL unit containing a buffering period SEI message (using that value of seq_parameter_set_id) that is not included in a scalable nesting SEI message and represents the last buffering period SEI message in decoding order in the access unit and DQIdMax is equal to 0, it is activated. This sequence parameter set RBSP is called the active SVC sequence parameter set RBSP until it is deactivated when another SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP. A sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

When a subset sequence parameter set RBSP (nal_unit_type is equal to 15) with a particular value of seq_parameter_set_id is not already the active SVC sequence parameter set RBSP and it is referred to by activation of a picture parameter set RBSP (using that value of seq_parameter_set_id) and the picture parameter set RBSP is activated

by a coded slice in scalable extension NAL unit (nal_unit_type is equal to 20) with DQId equal to DQIdMax (the picture parameter set RBSP becomes the active picture parameter set RBSP), it is activated. This subset sequence parameter set RBSP is called the active SVC sequence parameter set RBSP until it is deactivated when another SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP. A subset sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

When a subset sequence parameter set RBSP (nal_unit_type is equal to 15) with a particular value of seq_parameter_set_id is not already the active SVC sequence parameter set RBSP and it is referred to by an SEI NAL unit containing a buffering period SEI message (using that value of seq_parameter_set_id) that is included in a scalable nesting SEI message and represents the last buffering period SEI message in decoding order in the access unit that is associated with a value of DQId in the range of ((DQIdMax >> 4) << 4) to (((DQIdMax >> 4) << 4) + 15), inclusive, it is activated. This subset sequence parameter set RBSP, is called the active SVC sequence parameter set RBSP until it is deactivated when another SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP. A subset sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

NOTE 2 – The active SVC sequence parameter set RBSP is either a sequence parameter set RBSP or a subset sequence parameter set RBSP. Sequence parameter set RBSPs are activated by coded slice NAL units with nal_unit_type equal to 1 or 5 or buffering period SEI messages that are not included in a scalable nesting SEI message. Subset sequence parameter sets are activated by coded slice in scalable extension NAL units (nal_unit_type equal to 20) or buffering period SEI messages that are included in a scalable nesting SEI message. A sequence parameter set RBSP and a subset sequence parameter set RBSP may have the same value of seq_parameter_set_id.

For the following specification, the activating buffering period SEI message is specified as follows.

– If DQIdMax is less than 16 and the access unit contains one or more buffering period SEI messages and the last of these buffering period SEI messages in decoding order is not included in a scalable nesting SEI message, then this last buffering period SEI message is the activating buffering period SEI message.

– Otherwise, if the access unit contains one or more buffering period SEI messages that are included in a scalable nesting SEI message and are associated with a values of DQId in the range of ((DQIdMax >> 4) << 4) to (((DQIdMax >> 4) << 4) + 15), inclusive, then the last of these buffering period SEI messages in decoding order is the activating buffering period SEI message.

– Otherwise, the access unit does not contain an activating buffering period SEI message.

When an SVC sequence parameter set RBSP is referred to by activation of a picture parameter set RBSP inside a particular access unit and this picture parameter set RBSP is activated by a coded slice NAL unit with DQId equal to DQIdMax (the picture parameter set RBSP becomes the active picture parameter set RBSP) and this particular access unit also contains an activating buffering period SEI message that refers to an SVC sequence parameter set RBSP that is different then the SVC sequence parameter set RBSP referred to by the activation of the picture parameter set RBSP, then the SVC sequence parameter set RBSP that is referred to by the activating buffering period SEI message becomes the active SVC sequence parameter set.

NOTE 3 – Buffering period SEI messages have a higher priority for activating SVC sequence parameter sets then coded slice NAL units.

NOTE 4 – Compared to the specifications for profiles specified in Annex A, where an activated sequence parameter set RBSP must remain active for the entire coded video sequence, the specification for profiles specified in this annex differs. When an SVC sequence parameter set RBSP is already active (as the active SVC sequence parameter set RBSP), another SVC sequence parameter set RBSP becomes the active SVC sequence parameter set RBSP in a non-IDR access unit when it is referred to by an activating buffering period SEI message or by the activation of a picture parameter set RBSP (as the active picture parameter set RBSP). In this case, the contents of the de-activated and activated SVC sequence parameter set RBSP are mutually restricted as described below. Hence, within a coded video sequence, multiple successively activated/de-activated SVC sequence parameter set RBSPs can be present.

When a sequence parameter set RBSP (nal_unit_type is equal to 7) with a particular value of seq_parameter_set_id is not already the active layer SVC sequence parameter set RBSP for a particular value of DQId and it is referred to by activation of a picture parameter set RBSP (using that value of seq_parameter_set_id) and the picture parameter set RBSP is activated by a coded slice NAL unit with DQId less than DQIdMax (the picture parameter set RBSP becomes the active layer picture parameter set RBSP for the particular value of DQId) and nal_unit_type equal to 1 or 5, it is activated for the layer representation with the particular value of DQId. This sequence parameter set RBSP is called the active layer SVC sequence parameter set RBSP for the particular value of DQId until it is deactivated when another SVC sequence parameter set RBSP becomes the active layer SVC sequence parameter set RBSP for the particular value of DQId or when decoding an access unit with DQIdMax less than or equal to the particular value of DQId. A sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

When a subset sequence parameter set RBSP (nal_unit_type is equal to 15) with a particular value of seq_parameter_set_id is not already the active layer SVC sequence parameter set RBSP for a particular value of DQId and it is referred to by activation of a picture parameter set RBSP (using that value of seq_parameter_set_id) and the picture parameter set RBSP is activated by a coded slice NAL unit with DQId less than DQIdMax (the picture parameter set RBSP becomes the active layer picture parameter set RBSP for the particular value of DQId) and nal_unit_type equal to 20, it is activated for the layer representation with the particular value of DQId. This subset sequence parameter set RBSP is called the active layer SVC sequence parameter set RBSP for the particular value of DQId until it is deactivated when another SVC sequence parameter set RBSP becomes the active layer SVC sequence parameter set RBSP for the particular value of DQId or when decoding an access unit with DQIdMax less than or equal to the particular value of DQId. A subset sequence parameter set RBSP, with that particular value of seq_parameter_set_id, shall be available to the decoding process prior to its activation.

A sequence parameter set RBSP or a subset sequence parameter set RBSP that includes a value of profile_idc not specified in Annex A or Annex G shall not be referred to by activation of a picture parameter set RBSP as the active picture parameter set RBSP or as active layer picture parameter set RBSP (using that value of seq_parameter_set_id) or referred to by an SEI NAL unit containing a buffering period SEI message (using that value of seq_parameter_set_id). A sequence parameter set RBSP or a subset sequence parameter set RBSP including a value of profile_idc not specified in Annex A or Annex G is ignored in the decoding for profiles specified in Annex A or Annex G.

An SVC sequence parameter set RBSP (SPSB) that becomes the active SVC sequence parameter set RBSP in a non-IDR access unit is restricted with regards to its contents compared to the SVC sequence parameter set RBSP that it deactivates (SPSA) as specified in the following.

– The values of the syntax elements in the seq_parameter_set_data( ) syntax structure of SPSA and SPSB may only differ for the syntax elements profile_idc, constraint_setX_flag (with X being equal to 0 to 3, inclusive), level_idc, seq_parameter_set_id, and the hrd_parameters( ) syntax structure and shall be the same otherwise.

– When the seq_parameter_set_svc_extension( ) syntax structure is present in both SPSA and SPSB, the values of all syntax elements in the seq_parameter_set_svc_extension( ) syntax structure shall be the same.

For each particular value of DQId, the SVC sequence parameter set RBSPs that are active layer SVC sequence parameter set RBPSs for the particular value of DQId in any access unit inside a coded video sequence are restricted with regards to their contents. The values of the syntax elements in these SVC sequence parameter set RBSPs (for the particular value of DQId) may only differ for the syntax elements profile_idc, constraint_setX_flag (with X being equal to 0 to 3, inclusive), level_idc, and the hrd_parameters( ) syntax structure and shall be the same otherwise.

The active layer SVC sequence parameter set RBSPs for different values of DQId may be the same SVC sequence parameter set RBSP. The active SVC sequence parameter set RBSP and an active layer SVC sequence parameter set RBSP for a particular value of DQId may be the same SVC sequence parameter set RBSP.

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

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

For each particular value of DQId, the following applies.

– When the active layer SVC sequence parameter set RBSP is a sequence parameter set RBSP, any sequence parameter set RBSP with the value of seq_parameter_set_id for the active layer SVC sequence parameter set RBSP shall have the same content as that of the active layer SVC sequence parameter set RBSP unless it follows the last access unit of a coded video sequence and precedes the first VCL NAL unit of another coded video sequence.

– When the active layer SVC sequence parameter set RBSP is a subset sequence parameter set RBSP, any subset sequence parameter set RBSP with the value of seq_parameter_set_id for the active layer SVC sequence parameter set RBSP shall have the same content as that of the active layer SVC sequence parameter set RBSP unless it follows the last access unit of a coded video sequence and precedes the first VCL NAL unit of another coded video sequence.

NOTE 5 – If picture parameter set RBSP or sequence parameter set RBSP are conveyed within the bitstream, these constraints impose an order constraint on the NAL units that contain the picture parameter set RBSP or sequence parameter set RBSP,

respectively. Otherwise (picture parameter set RBSP or sequence parameter set RBSP are conveyed by other means not specified in this Recommendation | International Standard), they must be available to the decoding process in a timely fashion such that these constraints are obeyed.

When present, a sequence parameter set extension RBSP includes parameters having a similar function to those of a sequence parameter set RBSP. For purposes of establishing constraints on the syntax elements of the sequence parameter set extension RBSP and for purposes of determining activation of a sequence parameter set extension RBSP, the sequence parameter set extension RBSP shall be considered part of the preceding sequence parameter set RBSP with the same value of seq_parameter_set_id. When a sequence parameter set RBSP is present that is not followed by a sequence parameter set extension RBSP with the same value of seq_parameter_set_id prior to the activation of the sequence parameter set RBSP, the sequence parameter set extension RBSP and its syntax elements shall be considered not present for the active sequence parameter set RBSP. The contents of sequence parameter set extension RBSPs only apply when the base layer, which conforms to one or more of the profiles specified in Annex A, of a coded video sequence conforming to one or more profiles specified in Annex G is decoded. Subset sequence parameter set RBSPs shall not be followed by a sequence parameter set extension RBSP.

NOTE 6 – Sequence parameter sets extension RBSPs are not considered to be part of a subset sequence parameter set RBSP and subset sequence parameter set RBSPs shall not be followed by a sequence parameter set extension RBSP.

For layer representations with DQId equal to DQIdMax, all constraints that are expressed on the relationship between the values of the syntax elements (and the values of variables derived from those syntax elements) in SVC sequence parameter sets and picture parameter sets and other syntax elements are expressions of constraints that apply only to the active SVC sequence parameter set and the active picture parameter set. For layer representations with a particular value of DQId less than DQIdMax, all constraints that are expressed on the relationship between the values of the syntax elements (and the values of variables derived from those syntax elements) in SVC sequence parameter sets and picture parameter sets and other syntax elements are expressions of constraints that apply only to the active layer SVC sequence parameter set and the active layer picture parameter set for the particular value of DQId. If any SVC sequence parameter set RBSP having profile_idc equal to one of the profile_idc values specified in Annex A or Annex G is present that is never activated in the bitstream (i.e., it never becomes the active SVC sequence parameter set or an active layer SVC sequence parameter set), its syntax elements shall have values that would conform to the specified constraints if it were activated by reference in an otherwise-conforming bitstream. If any picture parameter set RBSP is present that is never ever activated in the bitstream (i.e., it never becomes the active picture parameter set or an active layer picture parameter set), its syntax elements shall have values that would conform to the specified constraints if it were activated by reference in an otherwise-conforming bitstream.

During operation of the decoding process (see clause G.8), for layer representations with DQId equal to DQIdMax, the values of parameters of the active picture parameter set and the active SVC sequence parameter set shall be considered in effect. For layer representations with a particular value of DQId less than DQIdMax, the values of the parameters of the active layer picture parameter set and the active layer SVC sequence parameter set for the particular value of DQId shall be considered in effect. For interpretation of SEI messages that apply to access unit or dependency representations with dependency_id equal to DependencyIdMax or layer representation with DQId equal to DQIdMax, the values of the parameters of the picture parameter set and SVC sequence parameter set that are the active picture parameter set and the active SVC sequence parameter set, respectively, for the operation of the decoding process for the VCL NAL units of the primary coded picture in the same access unit shall be considered in effect unless otherwise specified in the SEI message semantics. For interpretation of SEI messages that apply to dependency representations with a particular value of dependency_id less than DependencyIdMax, the values of the parameters of the picture parameter set and SVC sequence parameter set that are the active layer picture parameter set and the active layer SVC sequence parameter set, respectively, for the operation of the decoding process for the VCL NAL units of the layer representation with DQId equal to (dependency_id << 4) in the same access unit shall be considered in effect unless otherwise specified in the SEI message semantics. For interpretation of SEI messages that apply to layer representations with a particular value of DQId less than DQIdMax, the values of the parameters of the picture parameter set and SVC sequence parameter set that are the active layer picture parameter set and the active layer SVC sequence parameter set, respectively, for the operation of the decoding process for the VCL NAL units of the layer representation with the particular value of DQId in the same access unit shall be considered in effect unless otherwise specified in the SEI message semantics.

G.7.4.1.2.2 Order of access units and association to coded video sequences

The specification of subclause 7.4.1.2.2 applies with the following modifications.

The first access unit of the bitstream shall only contain coded slice NAL units with nal_unit_type equal to 5 or idr_flag equal to 1.

The order of NAL units and coded pictures and their association to access units is described in subclause G.7.4.1.2.3.

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

The specification of subclause 7.4.1.2.3 applies with the following modifications.

The association of VCL NAL units to primary or redundant coded pictures is specified in subclause G.7.4.1.2.5.

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

The constraint expressed in subclause 7.4.1.2.3 on the order of a buffering period SEI message is replaced by the following constraints.

- When an SEI NAL unit containing a buffering period SEI message is present, the following applies.
  - If the buffering period SEI message is the only buffering period SEI message in the access unit and it is not included in a scalable nesting SEI message, the buffering period SEI message shall be the first SEI message payload of the first SEI NAL unit in the access unit.
  - Otherwise (the buffering period SEI message is not the only buffering period SEI message in the access unit or it is included in a scalable nesting SEI message), the following applies.
    - When a buffering period SEI message that is not included in a scalable nesting SEI message is present, this buffering period SEI message shall be the only SEI message payload of the first SEI NAL unit in the access unit.
    - A scalable nesting SEI message that includes a buffering period SEI message shall not include any other SEI messages and the scalable nesting SEI message that includes a buffering period SEI message shall be the only SEI message inside an SEI NAL unit.
    - All SEI NAL units that precede an SEI NAL unit that contains a scalable nesting SEI message with a buffering period SEI message as payload in an access unit shall only contain buffering period SEI messages or scalable nesting SEI messages with a buffering period SEI message as payload.
    - When present, a scalable nesting SEI message with all_layer_representations_in_au_flag equal to 1 and a buffering period SEI message as payload shall be the first scalable nesting SEI message in an access unit.
    - Any scalable nesting SEI messages that precedes a particular scalable nesting SEI message with particular values of sei_dependency_id[i] and sei_quality_id[i] and a buffering period SEI message as payload shall have values (sei_dependency_id[i] << 4) + sei_quality_id[i] for all present i that are less than any of the particular values (sei_dependency_id[i] << 4) + sei_quality_id[i] for the particular scalable nesting SEI message with a buffering period SEI message as payload.

The following additional constraints shall be obeyed.

- Each NAL unit with nal_unit_type equal to 1, 5 or 12 shall be immediately preceded by a prefix NAL unit.
- Each prefix NAL unit shall be immediately followed by a NAL unit with nal_unit_type equal to 1, 5, or 12.

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

This subclause specifies constraints on VCL NAL unit syntax that are sufficient to enable the detection of the first VCL NAL unit of each primary coded picture.

The first VCL NAL unit of the primary coded picture of the current access unit, in decoding order, shall be different from the last VCL NAL unit of the primary coded picture of the previous access unit, in decoding order, in one or more of the following ways.

- dependency_id of the first VCL NAL unit of the primary coded picture of the current access unit is smaller than dependency_id of the last VCL NAL unit of the primary coded picture of the previous access unit
- dependency_id of the first VCL NAL unit of the primary coded picture of the current access unit is equal to dependency_id of the last VCL NAL unit of the primary coded picture of the previous access unit and any of the following conditions is true
  - quality_id of the first VCL NAL unit of the primary coded picture of the current access unit is smaller than quality_id of the last VCL NAL unit of the primary coded picture of the previous access unit
  - quality_id of the first VCL NAL unit of the primary coded picture of the current access unit and the last VCL NAL unit of the primary coded picture of the previous access unit is equal to 0, and any of the conditions specified in subclause 7.4.1.2.4 is fulfilled

G.7.4.1.2.5 Order of VCL NAL units and association to coded pictures

Each VCL NAL unit is part of a coded picture.

Let dId be the value of dependency_id and let qId be the value of quality_id of any particular VCL NAL unit. The order of the VCL NAL units within a coded picture is constrained as follows.

- For all VCL NAL units following this particular VCL NAL unit, the value of dependency_id shall be greater than or equal to dId.
- For all VCL NAL units with a value of dependency_id equal to dId following this particular VCL NAL unit, the value of quality_id shall be greater than or equal to qId.

For each set of VCL NAL units within a layer representation, the following applies.

- If arbitrary slice order, as specified in Annex A or subclause G.10, is allowed, coded slice NAL units of a layer representation may have any order relative to each other.
- Otherwise (arbitrary slice order is not allowed), coded slice NAL units of a slice group shall not be interleaved with coded slice NAL units of another slice group and the order of coded slice NAL units within a slice group shall be in the order of increasing macroblock address for the first macroblock of each coded slice NAL unit of the same slice group.

NAL units having nal_unit_type equal to 12 may be present in the access unit but shall not precede the first VCL NAL unit of the primary coded picture within the access unit.

NAL units having nal_unit_type equal to 0 or in the range of 24 to 31, inclusive, which are unspecified, may be present in the access unit but shall not precede the first VCL NAL unit of the primary coded picture within the access unit.

NAL units having nal_unit_type in the range of 21 to 23, inclusive, which are reserved, shall not precede the first VCL NAL unit of the primary coded picture within the access unit (when specified in the future by ITU-T | ISO/IEC).

G.7.4.2 Raw byte sequence payloads and RBSP trailing bits semantics

G.7.4.2.1 Sequence parameter set RBSP semantics

The semantics specified in subclause 7.4.2.1 apply.

G.7.4.2.1.1 Sequence parameter set data semantics

The semantics specified in subclause 7.4.2.1.1 apply with substituting SVC sequence parameter set for sequence parameter set. All constraints specified in subclause 7.4.2.1.1 apply only to the layer representations for which the SVC sequence parameter set is the active SVC sequence parameter set or the active layer SVC sequence parameter set as specified in subclause G.7.4.1.2.1. Additionally, the following applies.

constraint_set0_flag specifies the following.

- If the sequence parameter set data syntax structure is included in a sequence parameter set RBSP, the semantics specified in subclause 7.4.2.1.1 apply.
- Otherwise (the sequence parameter set data syntax structure is included in a subset sequence parameter set RBSP), contraint_set0_flag specifies that the coded video sequence obeys all constraints specified in subclause G.10.1.1. constraint_set0_flag equal to 0 specifies that the coded video sequence may or may not obey all constraints specified in subclause G.10.1.1.

constraint_set1_flag specifies the following.

- If the sequence parameter set data syntax structure is included in a sequence parameter set RBSP, the semantics specified in subclause 7.4.2.1.1 apply.
- Otherwise (the sequence parameter set data syntax structure is included in a subset sequence parameter set RBSP), contraint_set1_flag specifies that the coded video sequence obeys all constraints specified in subclause G.10.1.2. constraint_set1_flag equal to 0 specifies that the coded video sequence may or may not obey all constraints specified in subclause G.10.1.2.

constraint_set2_flag specifies the following.

- If the sequence parameter set data syntax structure is included in a sequence parameter set RBSP, the semantics specified in subclause 7.4.2.1.1 apply.
- Otherwise (the sequence parameter set data syntax structure is included in a subset sequence parameter set RBSP), the value of 1 for constraint_set2_flag is reserved for future use by ITU-T | ISO/IEC. constraint_set2_flag shall be equal to 0 for coded video sequences with profile_idc equal to 83 and 86 in bitstreams conforming to this

Recommendation | International Standard. Decoders conforming to this Recommendation | International Standard shall ignore the value of constraint_set2_flag when profile_idc is equal to 83 or 86.

constraint_set3_flag specifies the following.

- If the sequence parameter set data syntax structure is included in a sequence parameter set RBSP, the semantics specified in subclause 7.4.2.1.1 apply.
- Otherwise (the sequence parameter set data syntax structure is included in a subset sequence parameter set RBSP), the following applies.
  - If profile_idc is equal to 86, constraint_set3_flag equal to 1 specifies that the coded video sequence obeys all constraints specified in subclause G.10.1.3, and constraint_set3_flag equal to 0 specifies that the coded video sequence may or may not obey these corresponding constraints.
  - Otherwise (profile_idc is not equal to 86), the value of 1 for constraint_set3_flag is reserved for future use by ITU-T | ISO/IEC. constraint_set3_flag shall be equal to 0 for coded video sequences with profile_idc not equal to 86 in bitstreams conforming to this Recommendation | International Standard. Decoders conforming to this Recommendation | International Standard shall ignore the value of constraint_set3_flag when profile_idc is not equal to 86.

The value of separate_colour_plane_flag shall be equal to 0 and the value of qpprime_y_zero_transform_bypass_flag shall be equal to 0.

When the value of profile_idc does not indicate conformance to any of the profiles specified in Annex A and vui_parameters_present_flag is equal to 1, timing_info_present_flag shall be equal to 0, nal_hrd_parameters_present_flag shall be equal to 0, vcl_hrd_parameters_present_flag shall be equal to 0, and pic_struct_present_flag shall be equal to 0.

When the value of profile_idc does indicate conformance to one or more of the profiles specified in Annex A and vui_parameters_present_flag is equal to 1, the values of timing_info_present_flag, num_units_in_tick, time_scale, fixed_frame_rate_flag, nal_hrd_parameters_present_flag, vcl_hrd_parameters_present_flag, low_delay_hrd_flag, pic_struct_present_flag and the values of syntax elements included in the hrd_parameters( ) syntax structures, when present, shall be such that the bitstream activating the sequence parameter set is conforming to one or more of the profiles specified in Annex A.

G.7.4.2.1.1.1 Scaling list semantics

The semantics specified in subclause 7.4.2.1.1.1 apply.

G.7.4.2.1.2 Sequence parameter set extension RBSP semantics

The semantics specified in subclause 7.4.2.1.2 apply. Additionally, the following applies.

Sequence parameter set extension RBSPs can only follow sequence parameter set RBSPs in decoding order. Subset sequence parameter set RBSPs shall not be followed by a sequence parameter set extension RBSP. The contents of sequence parameter set extension RBSPs only apply when the base layer, which conforms to one or more of the profiles specified in Annex A, of a coded video sequence conforming to one or more profiles specified in Annex G is decoded.

G.7.4.2.1.3 Subset sequence parameter set RBSP semantics

svc_vui_parameters_present_flag equal to 0 indicates that the syntax structure svc_vui_parameters_extensions( ) is not present. svc_vui_parameters_present_flag equal to 1 indicates that the syntax structure svc_vui_parameters_extensions( ) is present.

additional_extension2_flag shall be equal to 0. The value of 1 for additional_extension2_flag is reserved for future use by ITU-T | ISO/IEC.

additional_extension2_data_flag may have any value. It shall not affect the conformance to profiles specified in Annex A or Annex G.

G.7.4.2.1.4 Sequence parameter set SVC extension semantics

inter_layer_deblocking_filter_control_present_flag equal to 1 specifies that a set of syntax elements controlling the characteristics of the deblocking filter for inter-layer prediction is present in the slice header. inter_layer_deblocking_filter_control_present_flag equal to 0 specifies that the set of syntax elements controlling the characteristics of the deblocking filter for inter-layer prediction is not present in the slice headers and their inferred values are in effect.

extended_spatial_scalability specifies the presence of syntax elements related to geometrical parameters for the resampling processes. The value of extended_spatial_scalability shall be in the range of 0 to 2, inclusive. When

extended_spatial_scalability is equal to 0, no geometrical parameters are present in the subset sequence parameter set and the slice headers referring to this subset sequence parameter set. When extended_spatial_scalability is equal to 1, geometrical parameters are present in the subset sequence parameter set, but not in the slice headers referring to this subset sequence parameter set. When extended_spatial_scalability is equal to 2, geometrical parameters are not present in the subset sequence parameter set, but they are present in the slice headers referring to this subset sequence parameter set.

chroma_phase_x_plus1_flag specifies the horizontal phase shift of the chroma components in units of half luma samples of a frame or layer frame. When chroma_phase_x_plus1_flag is not present, it shall be inferred to be equal to 1.

When chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are present, the following applies.

– If chroma_phase_x_plus1_flag is equal to 0, chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should be equal to 0, 2, or 4.

– Otherwise (chroma_phase_x_plus1_flag is equal to 1), chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should be equal to 1, 3, or 5.

chroma_phase_y_plus1 specifies the vertical phase shift of the chroma components in units of half luma samples of a frame or layer frame. When chroma_phase_y_plus1 is not present, it shall be inferred to be equal to 1. The value of chroma_phase_y_plus1 shall be in the range of 0 to 2, inclusive.

When chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field are present, the following applies.

– If chroma_phase_y_plus1 is equal to 0, chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should be equal to 2 or 3.

– Otherwise, if chroma_phase_y_plus1 is equal to 1, chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should be equal to 0 or 1.

– Otherwise (chroma_phase_y_plus1 is equal to 2), chroma_sample_loc_type_top_field and chroma_sample_loc_type_bottom_field should be equal to 4 or 5.

seq_ref_layer_chroma_phase_x_plus1_flag specifies the horizontal phase shift of the chroma components in units of half luma samples of a layer frame for the layer pictures that may be used for inter-layer prediction. When seq_ref_layer_chroma_phase_x_plus1_flag is not present, it shall be inferred to be equal to chroma_phase_x_plus1_flag.

seq_ref_layer_chroma_phase_y_plus1 specifies the vertical phase shift of the chroma components in units of half luma samples of a layer frame for the layer pictures that may be used for inter-layer prediction. When seq_ref_layer_chroma_phase_y_plus1 is not present, it shall be inferred to be equal to chroma_phase_y_plus1. The value of seq_ref_layer_chroma_phase_y_plus1 shall be in the range of 0 to 2, inclusive.

seq_scaled_ref_layer_left_offset specifies the horizontal offset between the upper-left luma sample of a resampled layer picture used for inter-layer prediction and the upper-left luma sample of the current picture or current layer picture in units of two luma samples. When seq_scaled_ref_layer_left_offset is not present, it shall be inferred to be equal to 0.

seq_scaled_ref_layer_top_offset specifies the vertical offset between the upper-left luma sample of an resampled layer picture used for inter-layer prediction and the upper-left luma sample of the current picture or current layer picture. The vertical offset is specified in units of two luma samples when frame_mbs_only_flag is equal to 1, and it is specified in units of four luma samples when frame_mbs_only_flag is equal to 0. When seq_scaled_ref_layer_top_offset is not present, it shall be inferred to be equal to 0.

seq_scaled_ref_layer_right_offset specifies the horizontal offset between the bottom-right luma sample of an resampled layer picture used for inter-layer prediction and the bottom-right luma sample of the current picture or current layer picture in units of two luma samples. When seq_scaled_ref_layer_right_offset is not present, it shall be inferred to be equal to 0.

seq_scaled_ref_layer_bottom_offset specifies the vertical offset between the bottom-right luma sample of an resampled layer picture used for inter-layer prediction and the bottom-right luma sample of the current picture or current layer picture. The vertical offset is specified in units of two luma samples when frame_mbs_only_flag is equal to 1, and it is specified in units of four luma samples when frame_mbs_only_flag is equal to 0. When seq_scaled_ref_layer_bottom_offset is not present, it shall be inferred to be equal to 0.

seq_tcoeff_level_prediction_flag specifies the presence of the syntax element adaptive_tcoeff_level_prediction_flag in the subset sequence parameter set.

adaptive_tcoeff_level_prediction_flag specifies the presence of tcoeff_level_prediction_flag in the slice headers that refer to the subset sequence parameter set. When adaptive_tcoeff_level_prediction_flag is not present, it shall be inferred to be equal to 0.

slice_header_restriction_flag specifies the presence of the syntax elements in the slice headers that refer to the subset sequence parameter set.

G.7.4.2.2 Picture parameter set RBSP semantics

The semantics specified in subclause 7.4.2.2 apply with substituting SVC sequence parameter set for sequence parameter set. All constraints specified in subclause 7.4.2.2 apply only to the layer representations for which the picture parameter set is the active picture parameter set or the active layer picture parameter set as specified in subclause G.7.4.1.2.1.

G.7.4.2.3 Supplemental enhancement information RBSP semantics

The semantics specified in subclause 7.4.2.3 apply.

G.7.4.2.3.1 Supplemental enhancement information message semantics

The semantics specified in subclause 7.4.2.3.1 apply.

G.7.4.2.4 Access unit delimiter RBSP semantics

The semantics specified in subclause 7.4.2.4 apply.

G.7.4.2.5 End of sequence RBSP semantics

The semantics specified in subclause 7.4.2.5 apply.

G.7.4.2.6 End of stream RBSP semantics

The semantics specified in subclause 7.4.2.6 apply.

G.7.4.2.7 Filler data RBSP semantics

The semantics specified in subclause 7.4.2.7 apply.

G.7.4.2.8 Slice layer without partitioning RBSP semantics

The semantics specified in subclause 7.4.2.8 apply.

G.7.4.2.9 Slice data partition RBSP semantics

Slice data partition syntax is not present in bitstreams conforming to one or more of the profiles specified in Annex G.

G.7.4.2.10 RBSP slice trailing bits semantics

The semantics specified in subclause 7.4.2.10 apply.

G.7.4.2.11 RBSP trailing bits semantics

The semantics specified in subclause 7.4.2.11 apply.

G.7.4.2.12 Prefix NAL unit RBSP semantics

store_ref_base_pic_flag equal to 1 specifies that, when the value of dependency_id as specified in the NAL unit header is equal to the maximum value of dependency_id for the VCL NAL units of the current access unit, an additional representation of the coded picture that may or may not be identical to the decoded picture is marked as "used for reference". This additional representation is also referred to as reference base picture and may be used for inter prediction of following pictures in decoding order, but it is not output. When store_ref_base_pic_flag is not present, it shall be inferred to be equal to 0.

The syntax element store_ref_base_pic_flag shall have an identical value for all VCL NAL units of a dependency representation. When nal_ref_idc is equal to 0, the value of store_ref_base_pic_flag shall be equal to 0.

store_ref_base_pic_flag is considered to apply to the associated NAL unit.

prefix_nal_unit_additional_extension_flag shall be equal to 0. The value of 1 for prefix_nal_unit_additional_extension_flag is reserved for future use by ITU-T | ISO/IEC.

prefix_nal_unit_extension_flag may have any value.

NOTE – The syntax elements prefix_nal_unit_additional_extension_flag and prefix_nal_unit_extension_flag are not required by the decoding process specified in this Recommendation | International Standard.

G.7.4.2.13 Slice layer in scalable extension RBSP semantics

The slice layer in scalable extension RBSP consists of a slice header in scalable extension and slice data in scalable extension.

G.7.4.3 Slice header semantics

The semantics specified in subclause 7.4.3 apply with the following modifications.

All constraints specified in subclause 7.4.3 apply only to the layer representations with DQId equal to 0.

frame_num has the same semantics as specified in subclause G.7.4.3.4.

The slice_header( ) syntax structure shall be considered to contain the following syntax elements with the following inferred values.

– ref_layer_dq_id is inferred to be equal to –1.

– scan_idx_start is inferred to be equal to 0.

– scan_idx_end is inferred to be equal to 15.

The value of direct_spatial_mv_pred_flag shall be equal to 1.

The variable SpatialResolutionChangeFlag is set equal to 0.

G.7.4.3.1 Reference picture list reordering semantics

The semantics specified in subclause 7.4.3.1 apply.

G.7.4.3.2 Prediction weight table semantics

The semantics specified in subclause 7.4.3.2 apply.

G.7.4.3.3 Decoded reference picture marking semantics

The semantics specified in subclause 7.4.3.3 apply with substituting SVC sequence parameter set for sequence parameter set. The constraints specified in subclause 7.4.3.3 apply only to the dependency representation with dependency_id equal to the current value of dependency_id and the modifications a) and b) specified in subclause G.8.2 apply with currDependencyId being equal to the current value of dependency_id.

G.7.4.3.4 Slice header in scalable extension semantics

The value of the following SVC sequence parameter set syntax elements shall be the same across all coded slice NAL units of an access unit: bit_depth_luma_minus8, bit_depth_chroma_minus8, and chroma_format_idc.

When present, the value of the following slice header syntax elements shall be the same across all coded slice NAL units of an access unit: field_pic_flag and bottom_field_flag.

When present, the value of the following slice header syntax elements shall be the same across all slices of a dependency representation including slice headers of NAL units with nal_unit_type equal to 1, 2, or 5: frame_num, idr_pic_id, pic_order_cnt_lsb, delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], and delta_pic_order_cnt[1].

When present, the value of the following slice header syntax elements shall be the same across all slices of a layer representation: pic_parameter_set_id, ref_layer_dq_id, disable_inter_layer_deblocking_filter_idc, inter_layer_slice_alpha_c0_offset_div2, inter_layer_slice_beta_offset_div2, constrained_intra_resampling_flag, ref_layer_chroma_phase_x_plus1_flag, ref_layer_chroma_phase_y_plus1, scaled_ref_layer_left_offset, scaled_ref_layer_top_offset, scaled_ref_layer_right_offset, scaled_ref_layer_bottom_offset, slice_group_change_cycle, store_ref_base_pic_flag, tcoeff_level_prediction_flag, and all syntax elements of the syntax structures dec_ref_pic_marking( ) and dec_ref_base_pic_marking( ).

Let setOfRefLayerSlices be the set of slices with dependency_id equal to the current value of dependency_id and quality_id equal to 0 that are covered or partly covered by the macroblocks of the current slice.

When quality_id is greater than 0, the value of (slice_type % 5) for all slices in the set setOfRefLayerSlices shall be the same as the value of (slice_type % 5) for the current slice.