**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.2.2 Picture timing SEI message semantics

NOTE 1 – The syntax of the picture timing SEI message is dependent on the content of the sequence parameter set that is active for the primary coded picture associated with the picture timing SEI message. However, unless the picture timing SEI message of an IDR access unit is preceded by a buffering period SEI message within the same access unit, the activation of the associated sequence parameter set (and, for IDR pictures that are not the first picture in the bitstream, the determination that the primary coded picture is an IDR picture) does not occur until the decoding of the first coded slice NAL unit of the primary coded picture. Since the coded slice NAL unit of the primary coded picture follows the picture timing SEI message in NAL unit order, there may be cases in which it is necessary for a decoder to store the RBSP containing the picture timing SEI message until determining the parameters of the sequence parameter that will be active for the primary coded picture, and then perform the parsing of the picture timing SEI message.

The presence of the 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 clause E.2.1) to wait after removal from the CPB of the access unit associated with the most recent buffering period SEI message in a preceding access unit 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 having a length in bits given by cpb\_removal\_delay\_length\_minus1 + 1. The cpb\_removal\_delay is the remainder of a modulo 2(cpb\_removal\_delay\_length\_minus1 + 1) counter.

NOTE 2 – The value of cpb\_removal\_delay\_length\_minus1 that determines the length (in bits) of the syntax element cpb\_removal\_delay is the value of cpb\_removal\_delay\_length\_minus1 coded in the sequence parameter set that is active for the primary coded picture associated with the picture timing SEI message, although cpb\_removal\_delay specifies a number of clock ticks relative to the removal time of the preceding access unit containing a buffering period SEI message, which may be an access unit of a different coded video sequence.

**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 clause C.2).

NOTE 3 – 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 4 – Only one dpb\_output\_delay is specified for a decoded picture.

The length 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 clause 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 in clauses 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 clause 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 5 – 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.

When pic\_struct is present (pic\_struct\_present\_flag is equal to 1), the constraints specified in the third column of Table D‑1 shall be obeyed.

NOTE 6 – When pic\_struct\_present\_flag is equal to 0, then in many cases default values may be inferred. In the absence of other indications of the intended display type of a picture, the decoder should infer the value of pic\_struct as follows:

– If field\_pic\_flag is equal to 1, pic\_struct should be inferred to be equal to (1 + bottom\_field\_flag).

– Otherwise, if TopFieldOrderCnt is equal to BottomFieldOrderCnt, pic\_struct should be inferred to be equal to 0.

– Otherwise, if TopFieldOrderCnt is less than BottomFieldOrderCnt, pic\_struct should be inferred to be equal to 3.

– Otherwise (field\_pic\_flag is equal to 0 and TopFieldOrderCnt is greater than BottomFieldOrderCnt), pic\_struct should be inferred to be equal to 4.

pic\_struct is only a hint as to how the decoded video should be displayed on an assumed display type (e.g., interlaced or progressive) at an assumed display rate. When another display type or display rate is used by the decoder, then pic\_struct does not indicate the display method, but may aid in processing the decoded video for the alternative display. When it is desired for pic\_struct to have an effective value in the range of 5 to 8, inclusive, pic\_struct\_present\_flag should be equal to 1, as the above inference rule will not produce these values.

Table D‑1 – Interpretation of pic\_struct

|  |  |  |  |
| --- | --- | --- | --- |
| **Value** | **Indicated display of picture** | **Restrictions** | **NumClockTS** |
| 0 | (progressive) frame | field\_pic\_flag shall be 0, TopFieldOrderCnt shall be equal to BottomFieldOrderCnt | 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, TopFieldOrderCnt shall be less than or equal to BottomFieldOrderCnt | 2 |
| 4 | bottom field, top field, in that order | field\_pic\_flag shall be 0, BottomFieldOrderCnt shall be less than or equal to TopFieldOrderCnt | 2 |
| 5 | top field, bottom field, top field repeated, in that order | field\_pic\_flag shall be 0, TopFieldOrderCnt shall be less than or equal to BottomFieldOrderCnt | 3 |
| 6 | bottom field, top field, bottom field repeated, in that order | field\_pic\_flag shall be 0, BottomFieldOrderCnt shall be less than or equal to TopFieldOrderCnt | 3 |
| 7 | frame doubling | field\_pic\_flag shall be 0, fixed\_frame\_rate\_flag shall be 1, TopFieldOrderCnt shall be equal to BottomFieldOrderCnt | 2 |
| 8 | frame tripling | field\_pic\_flag shall be 0, fixed\_frame\_rate\_flag shall be 1, TopFieldOrderCnt shall be equal to BottomFieldOrderCnt | 3 |
| 9..15 | reserved |  |  |

When fixed\_frame\_rate\_flag is equal to 1, it is a requirement of bitstream conformance that the constraints specified as follows shall be obeyed throughout the operation of the following process, which is operated in output order.

1. Prior to output of the first picture of the bitstream (in output order) and prior to the output of the first picture (in output order) of each subsequent coded video sequence for which the content of the active sequence parameter set differs from that of the previously-active sequence parameter set, the variable lastFieldBottom is set equal to "not determined".
2. After the output of each picture, the value of lastFieldBottom is checked and set as follows, using the values of field\_pic\_flag, bottom\_field\_flag, pic\_struct, TopFieldOrderCnt and BottomFieldOrderCnt (when applicable) for the picture that was output.

– If field\_pic\_flag is equal to 1, it is a requirement of bitstream conformance that the value of lastFieldBottom shall not be equal to bottom\_field\_flag. The value of lastFieldBottom is then set equal to bottom\_field\_flag.

– Otherwise (field\_pic\_flag is equal to 0), the following applies:

– If pic\_struct is present and is equal to 3 or 5, it is a requirement of bitstream conformance that the value of lastFieldBottom shall not be equal to 0. The value of lastFieldBottom is then set equal to 1 − ( ( pic\_struct − 1 ) >> 2 ).

– Otherwise, if pic\_struct is present and is equal to 4 or 6, it is a requirement of bitstream conformance that the value of lastFieldBottom shall not be equal to 1. The value of lastFieldBottom is then set equal to ( ( pic\_struct − 1 ) >> 2 ).

– Otherwise, if TopFieldOrderCnt is less than BottomFieldOrderCnt, it is a requirement of bitstream conformance that the value of lastFieldBottom shall not be equal to 0. The value of lastFieldBottom is then set equal to 1.

– Otherwise, if TopFieldOrderCnt is greater than BottomFieldOrderCnt, it is a requirement of bitstream conformance that the value of lastFieldBottom shall not be equal to 1. The value of lastFieldBottom is then set equal to 0.

– Otherwise (TopFieldOrderCnt is equal to BottomFieldOrderCnt and pic\_struct is not present or is not in the range of 3 to 6, inclusive), lastFieldBottom may have any value, and its value is not changed.

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

clockTimestamp = ( ( hH \* 60 + mM ) \* 60 + sS ) \* time\_scale +   
 nFrames \* ( num\_units\_in\_tick \* ( 1 + nuit\_field\_based\_flag ) ) + tOffset, (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 7 – 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 ÷ ( 2 \* num\_units\_in\_tick ) ) (D-2)

NOTE 8 – 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 9 – 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 50 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 10 – 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 50 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 − 1 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 sSshall 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 11 – 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 values of n\_frames equal to 0 and 1 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 12 – 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 sSto 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.

**E.1.2 HRD parameters**

|  |  |  |
| --- | --- | --- |
| hrd\_parameters( ) { | C | Descriptor |
| **cpb\_cnt\_minus1** | 0 | 5 | ue(v) |
| **bit\_rate\_scale** | 0 | 5 | u(4) |
| **cpb\_size\_scale** | 0 | 5 | u(4) |
| for( SchedSelIdx = 0; SchedSelIdx <= cpb\_cnt\_minus1; SchedSelIdx++ ) { |  |  |
| **bit\_rate\_value\_minus1[** SchedSelIdx **]** | 0 | 5 | ue(v) |
| **cpb\_size\_value\_minus1[** SchedSelIdx **]** | 0 | 5 | ue(v) |
| **cbr\_flag[** SchedSelIdx **]** | 0 | 5 | u(1) |
| } |  |  |
| **initial\_cpb\_removal\_delay\_length\_minus1** | 0 | 5 | u(5) |
| **cpb\_removal\_delay\_length\_minus1** | 0 | 5 | u(5) |
| **dpb\_output\_delay\_length\_minus1** | 0 | 5 | u(5) |
| **time\_offset\_length** | 0 | 5 | u(5) |
| } |  |  |

**E.2.2 HRD parameters semantics**

[..]

**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.