|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  16th Meeting: San José, US, 9–17 Jan. 2014 | Document: JCTVC-P0261 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Extension of the pic\_struct element in HEVC** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Alexis Michael Tourapis  David Singer  Krasimir Kolarov  Steve Saunders  1 Infinite Loop, Cupertino, CA, 95014, U.S.A | Email: | atourapis@apple.com |
| *Source:* | Apple Inc. | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

In this contribution, an extension of the pic\_struct element that is present in the Picture timing SEI message of HEVC is proposed. This extension enables carriage of interlaced video material using a “progressive”-like carriage mechanism that does not require any other coding tool modifications in the specification.



Figure . Encoding of Interlace Material using a Progressive Carriage mechanism.

# Introduction

It has been suggested that interlaced material can be carried using the HEVC specification using a progressive-like carriage mechanism. That is, an interlace sequence can first, and in one sense "temporarily, be de-interlaced using any available de-interlacing strategy, encoded using progressive video coding tools (i.e. using the HEVC Main or Main 10 profiles), and when decoded, the decoder may select to “re-interlace” the material back into its original interlace form for further processing if desired (Figure 1). This method has the advantage that it does not require any new coding tool additions within a video codec such as HEVC for the support of interlace material, while, if it is desired, the content can be kept in a progressive mode for further consumption. Performing deinterlacing at the source can have substantial benefits in multiple scenarios since deinterlacing at the source could likely produce substantially better deinterlacing results (given the likely higher quality of the source and reduced coding artifacts), while reducing complexity at the decoder in terms of having to provide and use a de-interlacing scheme for display. On the other hand, slightly more complexity is needed at the encoder and decoder since now one likely needs to process and encode/decode twice the samples. In general, and depending on the deinterlacing scheme used, some coding benefits could be achieved also using this strategy given the better coding characteristics of progressive sources compared to interlace ones and given the availability of coding tools within codecs such as HEVC (square transforms with symmetries in the scanning order of coefficients, phases of interpolation filters etc).

Although we believe that this kind of strategy would be highly desirable and could work within existing deployed systems and solutions, one limitation exists in the fact that systems that wish to be aware of the interlace nature of the original content have currently no mechanism to know so.

We therefore propose a very simple extension of the pic\_struct element that is currently available in the Picture timing SEI message of HEVC so as to enable signalling of whether the current picture is a deinterlaced picture, and which samples in the picture correspond to the original interlace data. In particular, we propose introducing pic\_struct values of 13 and 14, which indicate that a picture consists of a top field paired with its interpolated bottom field data, and a bottom field paired with its interpolated top field data respectively. By encountering this information a decoder can decide which data to discard, if it so desires, to reconstruct back the source in its interlace format.

It should be noted that using this approach no special handling needs to be performed for 4:2:2 material. However, for 4:2:0 content, chroma samples would have to be appropriately converted when deinterlacing or reinterlacing to accommodate for differences in phase commonly encountered between chroma samples in the interlace and progressive formats. Nevertheless, it is not the intention of this document to specify how this is best to be performed.

# Conclusions

In this contribution, an extension of the values supported by the pic\_struct element in the Picture Timing SEI message of HEVC was proposed. With this extension it is now possible to deinterlace an interlace video source, encode it with HEVC, and signal to appropriate decoders how to appropriately re-interlace the source if it is so desired. .

# Patent rights declaration(s)

**Apple Inc. does not have any current or pending patent rights relating to the technology described in this contribution.**

# Spec text modification

**D3.3** **Picture timing SEI message semantics**

**pic\_struct** indicates whether a picture should be displayed as a frame or as one or more fields. In addition, for the display of frames when fixed\_pic\_rate\_within\_cvs\_flag is equal to 1, it may indicate a frame doubling or tripling repetition period for displays that use a fixed frame refresh interval equal to DpbOutputElementalInterval[ n ] as given by Equation E 51. On the other hand, for the display of fields it may indicate a field to frame interpolation which may require discarding half of the samples for display. The interpretation of pic\_struct is specified in Table D 2. Values of pic\_struct that are not listed in Table D 2 are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders shall ignore reserved values of pic\_struct.

When present, it is a requirement of bitstream conformance that the value of pic\_struct shall be constrained such that exactly one of the following conditions is true:

– The value of pic\_struct is equal to 0, 7, or 8 for all pictures in the CVS.

– The value of pic\_struct is equal to 1, 2, 9, 10, 11, or 12 for all pictures in the CVS.

– The value of pic\_struct is equal to 3, 4, 5, or 6 for all pictures in the CVS.

– The value of pic\_struct is equal to 13 or 14 for all pictures in the CVS.

When fixed\_pic\_rate\_within\_cvs\_flag is equal to 1, frame doubling is indicated by pic\_struct equal to 7, which indicates that the frame should be displayed two times consecutively on displays with a frame refresh interval equal to DpbOutputElementalInterval[ n ] as given by Equation E 51, and frame tripling is indicated by pic\_struct equal to 8, which indicates that the frame should be displayed three times consecutively on displays with a frame refresh interval equal to DpbOutputElementalInterval[ n ] as given by Equation E 51.

NOTE 2 – Frame doubling can be used to facilitate the display, for example, of 25 Hz progressive-scan video on a 50 Hz progressive-scan display or 30 Hz progressive-scan video on a 60 Hz progressive-scan display. Using frame doubling and frame tripling in alternating combination on every other frame can be used to facilitate the display of 24 Hz progressive-scan video on a 60 Hz progressive-scan display.

The nominal vertical and horizontal sampling locations of samples in top and bottom fields for 4:2:0, 4:2:2, and 4:4:4 chroma formats are shown in Figure D 1, Figure D 2, and Figure D 3, respectively.

Association indicators for fields (pic\_struct equal to 9 through 12) provide hints to associate fields of complementary parity together as frames. The parity of a field can be top or bottom, and the parity of two fields is considered complementary when the parity of one field is top and the parity of the other field is bottom.

When frame\_field\_info\_present\_flag is equal to 1, it is a requirement of bitstream conformance that the constraints specified in the third column of Table D 2 shall apply.

NOTE 3 – When frame\_field\_info\_present\_flag is equal to 0, then in many cases default values may be inferred or indicated by other means. In the absence of other indications of the intended display type of a picture, the decoder should infer the value of pic\_struct as equal to 0 when frame\_field\_info\_present\_flag is equal to 0.

When pic\_struct is equal to 13 or 14 that indicates that the current picture, which is in frame mode, has been interpolated using some kind of de-interlacing scheme, and the field data indicated by this SEI message (top for 13, bottom for 14) correspond to the same parity data present in the original source. This information can be used to then extract only these samples for future processing, if desired, and discard all other samples.

Table D‑2 – Interpretation of pic\_struct

|  |  |  |
| --- | --- | --- |
| **Value** | **Indicated display of picture** | **Restrictions** |
| 0 | (progressive) frame | field\_seq\_flag shall be 0 |
| 1 | top field | field\_seq\_flag shall be 1 |
| 2 | bottom field | field\_seq\_flag shall be 1 |
| 3 | top field, bottom field, in that order | field\_seq\_flag shall be 0 |
| 4 | bottom field, top field, in that order | field\_seq\_flag shall be 0 |
| 5 | top field, bottom field, top field repeated, in that order | field\_seq\_flag shall be 0 |
| 6 | bottom field, top field, bottom field repeated, in that order | field\_seq\_flag shall be 0 |
| 7 | frame doubling | field\_seq\_flag shall be 0 fixed\_pic\_rate\_within\_cvs\_flag shall be 1 |
| 8 | frame tripling | field\_seq\_flag shall be 0 fixed\_pic\_rate\_within\_cvs\_flag shall be 1 |
| 9 | top field paired with previous bottom field in output order | field\_seq\_flag shall be 1 |
| 10 | bottom field paired with previous top field in output order | field\_seq\_flag shall be 1 |
| 11 | top field paired with next bottom field in output order | field\_seq\_flag shall be 1 |
| 12 | bottom field paired with next top field in output order | field\_seq\_flag shall be 1 |
| 13 | Interleaved frame with bottom field interpolation | field\_seq\_flag shall be 0 |
| 14 | Interleaved frame with top  field interpolation | field\_seq\_flag shall be 0 |

E.2.2 HRD parameters semantics

Divisor for computation of DpbOutputElementalInterval[ n ]

|  |  |  |
| --- | --- | --- |
| **frame\_field\_info\_present\_flag** | **pic\_struct** | **DeltaToDivisor** |
| 0 | - | 1 |
| 1 | 1 | 1 |
| 1 | 2 | 1 |
| 1 | 0 | 2 |
| 1 | 3 | 2 |
| 1 | 4 | 2 |
| 1 | 5 | 3 |
| 1 | 6 | 3 |
| 1 | 7 | 2 |
| 1 | 8 | 3 |
| 1 | 9 | 1 |
| 1 | 10 | 1 |
| 1 | 11 | 1 |
| 1 | 12 | 1 |
| 1 | 13 | 2 |
| 1 | 14 | 2 |