|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  9th Meeting: Geneva, CH, 27 April – 7 May 2012 | Document: JCTVC-I0260 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **On Weighted Prediction Parameter Signalling** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Yong He, Jie Dong, Yan Ye, Eun Seok Ryu,  9710 Scranton Rd, #250 San Diego, CA 92121 | Tel: Email: | +1-858-210-4803 [Yong.he@interdigital.com](mailto:Yong.he@interdigital.com)  [jie.dong@interdigital.com](mailto:jie.dong@interdigital.com)  [yan.ye@interdigital.com](mailto:yan.ye@interdigital.com)  [eun.ryu@interdigital.com](mailto:eun.ryu@interdigital.com) |
| *Source:* | InterDigital Communications, LLC | | |

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

# Abstract

In the current HEVC CD [1], the explicit Weighted Prediction parameters in B slices are signalled either for the L0 and L1, or for the combined list LC. Switching of these two types of signalling is controlled by the flag ref\_pic\_list\_combination\_flag. In this contribution, a revised WP parameter signalling method is proposed with the goal to fix problems in the current scheme and to simplify the syntax design for WP parameter signalling.

# Problem statement

At the 7th JCT-VC meeting, redundancy reduction of WP parameter signalling [4] was adopted to combine the signalling of explicit WP parameters for different reference picture lists in B slices. Table 1 shows the current syntax table used to signal the explicit WP parameters in the slice header. When ref\_pic\_list\_combination\_flag is 1, WP parameters are signalled according to the LC indices, and the weights and offsets used in bi-prediction are derived by mapping a given L0 or L1 index to the corresponding LC index and retrieving the WP parameters sent for that LC index. When ref\_pic\_list\_combination\_flag is 0, WP parameters are signalled for L0 and L1. Drawbacks of the current scheme include the following:

1. The lists combination LC is a “virtual” list mapped from L0 and L1. It is therefore somewhat “unnatural” to carry WP parameters using LC and map them back to L0 and L1 for them to be used. For example, if some pictures on L0 and L1 are the same (such as in LD case), but the encoder wants to assign different WP parameters, the encoder must set ref\_pic\_list\_combination\_flag = 1 and ref\_pic\_list\_modification\_flag\_lc = 1. The encoder then must signal each entry in LC explicitly before it can signal the WP parameters. This incurs additional bit overhead at the slice level. Furthermore, because the LC size becomes larger than necessary, this increases cost of sending ref\_idx\_lc at PU level.
2. The list combination syntax (see Table 2) does not mandate that all L0 and L1 indices must be mapped to an LC index. Consequently, the current signalling method could result in “missing” WP parameters for some references. For example, in Figure 1, current picture POC = 10. In random access setting, L0 and L1 each contains 2 entries (num\_ref\_idx\_l0\_active\_minus1 = num\_ref\_idx\_l1\_active\_minus1 = 1). In Table 2, if LC is configured to only contain 2 entries (num\_ref\_idx\_lc\_active\_minus1 = 1), then no WP parameters can be sent for reference pictures with POC = 6 in L0 and POC = 16 in L1. For these reference pictures, the default parameters must be used, and the text in [1] is not properly written for this case.
3. When ref\_pic\_list\_combination\_flag is 0, then L0 and L1 must be identical by definition. However, in the current scheme, according to Table 1, WP parameters are sent for both L0 and L1. This leads to additional signalling redundancy.
4. The decoding process in 8.5.2.2.3.2 weighted sample prediction process uses the array RefIdxLCToRefIdxLx[ refIdxL0/refIdxL1] to map L0 or L1 index to LC index and to find the corresponding weighting parameters, e.g., w0 = LumaWeightLC[RefIdxLCToRefIdxLx[ refIdxL0]]. However, this mapping is incorrect, since RefIdxLCToRefIdxLx[ ] maps LC index to an L0 or L1 index, not in the reverse direction. Additional arrays must be defined in order to make the current decoding process in 8.5.2.2.3.2 work correctly.



Figure . Example of inability to provide WP parameters for all pictures in L0 and L1 using current method

Table . WP parameter signaling in [1]

|  |  |
| --- | --- |
| pred\_weight\_table( ) { | Descriptor |
| **luma\_log2\_weight\_denom** | ue(v) |
| if( chroma\_format\_idc != 0 ) |  |
| **delta\_chroma\_log2\_weight\_denom** | se(v) |
| if( slice\_type = = P | |  ( slice\_type = = B && ref\_pic\_list\_combination\_flag = = 0 ) ) { |  |
| for( i = 0; i <= num\_ref\_idx\_l0\_active\_minus1; i++ ) { |  |
| **luma\_weight\_l0\_flag** | u(1) |
| if( luma\_weight\_l0\_flag ) { |  |
| **delta\_luma\_weight\_l0[** i **]** | se(v) |
| **luma\_offset\_l0[** i **]** | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l0\_flag** | u(1) |
| if( chroma\_weight\_l0\_flag ) |  |
| for( j =0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l0[** i **][** j **]** | se(v) |
| **delta\_chroma\_offset\_l0[** i **][** j **]** | se(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| if( slice\_type = = B ) { |  |
| if( ref\_pic\_list\_combination\_flag = = 0 ) { |  |
| for( i = 0; i <= num\_ref\_idx\_l1\_active\_minus1; i++ ) { |  |
| **luma\_weight\_l1\_flag** | u(1) |
| if( luma\_weight\_l1\_flag ) { |  |
| **delta\_luma\_weight\_l1[** i **]** | se(v) |
| **luma\_offset\_l1[** i **]** | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l1\_flag** | u(1) |
| if( chroma\_weight\_l1\_flag ) |  |
| for( j = 0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l1[** i **][** j **]** | se(v) |
| **delta\_chroma\_offset\_l1[** i **][** j **]** | se(v) |
| } |  |
| } |  |
| } |  |
| } else { |  |
| for( i = 0; i <= num\_ref\_idx\_lc\_active\_minus1; i++ ) { |  |
| **luma\_weight\_lc\_flag** | u(1) |
| if( luma\_weight\_l1\_flag ) { |  |
| **delta\_luma\_weight\_lc[** i **]** | se(v) |
| **luma\_offset\_lc[** i **]** | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_lc\_flag** | u(1) |
| if( chroma\_weight\_lc\_flag ) |  |
| for( j = 0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_lc[** i **][** j **]** | se(v) |
| **delta\_chroma\_offset\_lc[** i **][** j **]** | se(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |

Table . Reference picture list combination syntax in [1]

|  |  |
| --- | --- |
| ref\_pic\_list\_combination( ) { | Descriptor |
| if( slice\_type = = B ) { |  |
| **ref\_pic\_list\_combination\_flag** | u(1) |
| if( ref\_pic\_list\_combination\_flag ) { |  |
| **num\_ref\_idx\_lc\_active\_minus1** | ue(v) |
| **ref\_pic\_list\_modification\_flag\_lc** | u(1) |
| if( ref\_pic\_list\_modification\_flag\_lc) |  |
| for( i =0; i <= num\_ref\_idx\_lc\_active\_minus1; i++ ) { |  |
| **pic\_from\_list\_0\_flag** | u(1) |
| if( ( pic\_from\_list\_0\_flag &&  num\_ref\_idx\_l0\_active\_minus1 > 0 ) | |  ( !pic\_from\_list\_0\_flag &&  num\_ref\_idx\_l1\_active\_minus1 > 0 ) |  |
| **ref\_idx\_list\_curr** | ue(v) |
| } |  |
| } |  |
| } |  |
| } |  |

# Proposed weighted prediction signaling

Instead of having two options of sending WP parameters (for LC or for L0 and L1) based on the value of ref\_pic\_list\_combination\_flag, it is proposed here to always signal WP parameters for L0 and L1. For blocks that are single-list predicted using refIdxLC in the combined list LC, the WP parameters are obtained by inheriting from those of the corresponding entry in L0 or L1, depending on the values of PredLCToPredLx[ refIdxLC ] and RefIdxLCToRefIdxLx[ refIdxLC ]. The rest of the inter prediction process already uses these two arrays to identify reference pictures used for PU prediction, which makes the WP decoding process more coherent with the rest of the specification.

Directly signaling WP parameters for L0 or L1 can introduce additional signaling overhead when some reference pictures in L0 and L1 are the same. An example is given in Figure 2, where the L0, L1 and LC entries for POC=12 in random access setting in common testing conditions [2] are shown. Since ref POC = 8 appears on both L0 and L1, signaling the WP parameters twice incurs additional bit overhead. This problem is even bigger for the low delay settings in [2], as L0 and L1 are completely identical in LD case. To reduce signaling overhead, when L0 and L1 are identical (ref\_pic\_list\_combination\_flag = 0), an additional flag, weights\_l1\_present\_flag is sent to indicate if any WP parameters for L1 entries are signaled separately. Additionally, if WP parameters for a given reference picture has been sent previously, the WP parameters for the same reference picture are predicted from those already sent. In the example of Figure 2, the WP parameters of refIdxL1 = 1 (ref pic POC=8) of L1 are predicted from the WP parameters of refIdx0 = 0 of L0 (also ref pic POC = 8) previously sent to reduce signaling overhead.



Figure . Redundant signaling of WP parameters when L0 and L1 have overlapping entries

In summary, Table 3 is the proposed syntax table for pred\_weight\_table(). Benefits of the proposed scheme include the following:

1. The proposed method fixes the problems in the current design as discussed above.
2. The proposed method reduces the size of syntax table pred\_weight\_table().
3. Reference pictures on L0 and L1 can carry the same or different WP parameters, even when some of them refer to the same physical reference picture in the DPB. This allows the encoder additional flexibility when deriving the optimal WP parameters.
4. The proposed scheme can be used together with reference picture reordering to allow the reference pictures on LC to carry different WP parameters. Since LC references are used in uni-prediction whereas L0/L1 references are used in bi-prediction, allowing them to have different WP parameters can bring performance benefits.

Table . Proposed WP parameter signaling

|  |  |
| --- | --- |
| pred\_weight\_table( ) { | Descriptor |
| **luma\_log2\_weight\_denom** | ue(v) |
| if( chroma\_format\_idc != 0 ) |  |
| **delta\_chroma\_log2\_weight\_denom** | se(v) |
| ~~if( slice\_type = = P | |~~  ~~( slice\_type = = B && ref\_pic\_list\_combination\_flag = = 0 ) ) {~~ |  |
| for( i = 0; i <= num\_ref\_idx\_l0\_active\_minus1; i++ ) { |  |
| **luma\_weight\_l0\_flag** [ i ] | u(1) |
| if( luma\_weight\_l0\_flag [ i ]) { |  |
| **delta\_luma\_weight\_l0**[ i ] | se(v) |
| ~~luma\_offset\_l0[ i ]~~ **delta\_luma\_offset\_l0**[ i ] | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l0\_flag [i]** | u(1) |
| if( chroma\_weight\_l0\_flag [i] ) |  |
| for( j =0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l0**[ i ][ j ] | se(v) |
| **delta\_chroma\_offset\_l0**[ i ][ j ] | se(v) |
| } |  |
| } |  |
| } |  |
| ~~}~~ |  |
| if( slice\_type = = B ) { |  |
| if( ref\_pic\_list\_combination\_flag = = 0 ) // if L0 and L1 are identical |  |
| **weights\_l1\_present\_flag** | u(1) |
| if( ~~ref\_pic\_list\_combination\_flag = = 0~~ weights\_l1\_present\_flag) { |  |
| for( i = 0; i <= num\_ref\_idx\_l1\_active\_minus1; i++ ) { |  |
| **delta\_params\_present\_flag [i]** | u(1) |
| if (delta\_params\_present\_flag [i]) { |  |
| **luma\_weight\_l1\_flag** [ i ] | u(1) |
| if( luma\_weight\_l1\_flag[ i ]) { |  |
| **delta\_luma\_weight\_l1**[ i ] | se(v) |
| ~~luma\_offset\_l1[ i ]~~ **delta\_luma\_offset\_l1**[ i ] | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l1\_flag**[ i ] | u(1) |
| if( chroma\_weight\_l1\_flag [ i ]) |  |
| for( j = 0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l1**[ i ][ j ] | se(v) |
| **delta\_chroma\_offset\_l1**[ i ][ j ] | se(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } ~~else {~~ |  |
| ~~for( i = 0; i <= num\_ref\_idx\_lc\_active\_minus1; i++ ) {~~ |  |
| **~~luma\_weight\_lc\_flag~~** | ~~u(1)~~ |
| ~~if( luma\_weight\_lc\_flag ) {~~ |  |
| **~~delta\_luma\_weight\_lc~~**~~[ i ]~~ | ~~se(v)~~ |
| ~~luma\_offset\_lc[ i ]~~ | ~~se(v)~~ |
| ~~}~~ |  |
| ~~if( chroma\_format\_idc != 0 ) {~~ |  |
| **~~chroma\_weight\_lc\_flag~~** | ~~u(1)~~ |
| ~~if( chroma\_weight\_lc\_flag)~~ |  |
| ~~for( j = 0; j < 2; j++ ) {~~ |  |
| **~~delta\_chroma\_weight\_lc~~**~~[ i ][ j ]~~ | ~~se(v)~~ |
| **~~delta\_chroma\_offset\_lc~~**~~[ i ][ j ]~~ | ~~se(v)~~ |
| ~~}~~ |  |
| ~~}~~ |  |
| ~~}~~ |  |
| ~~}~~ |  |
| } |  |
| } |  |

# Proposed weighted prediction semantics

**luma\_log2\_weight\_denom** is the base 2 logarithm of the denominator for all luma weighting factors. The value of luma\_log2\_weight\_denom shall be in the range of 0 to 7, inclusive.

**delta\_chroma\_log2\_weight\_denom** is the difference of the base 2 logarithm of the denominator for all chroma weighting factors.

The variable ChromaLog2WeightDenom is specified by luma\_log2\_weight\_denom + delta\_chroma\_log2\_weight\_denom and it shall be in the range of 0 to 7, inclusive.

**luma\_weight\_l0\_flag[** i **]** equal to 1 specifies that weighting factors for the luma component of list 0 prediction using RefPicList0[ i ] are present. luma\_weight\_l0\_flag[ i ] equal to 0 specifies that these weighting factors are not present.

**delta\_luma\_weight\_l0[** i **]** is the difference of the weighting factor applied to the luma prediction value for list 0 prediction using RefPicList0[ i ].

The variable LumaWeightL0[ i ] is specified ~~by (1 << luma\_log2\_weight\_denom ) + delta\_luma\_weight\_l0[ i ]~~ by invoking the process in 8.3.7. ~~When luma\_weight\_l0\_flag is equal to 1,~~ The value of LumaWeightL0[ i ] shall be in the range of −128 to 127, inclusive. ~~When luma\_weight\_l0\_flag~~~~is equal to 0, LumaWeightL0[ i ] is inferred to be equal to 2~~~~luma\_log2\_weight\_denom~~ ~~for RefPicList0[ i ].~~

**~~luma\_offset\_l0[~~**~~i~~**~~]~~** **delta\_luma\_offset\_l0**[ i ] is the difference of the additive offset applied to the luma prediction value for list 0 prediction using RefPicList0[ i ]. The variable LumaOffsetL0[ i ] is specified by invoking the process in 8.3.7. The value of ~~luma\_offset\_l0~~LumaOffsetL0[ i ] shall be in the range of −128 to 127, inclusive. ~~When luma\_weight\_l0\_flag~~~~is equal to 0, luma\_offset\_l0LumaOffsetL0[ i ] is inferred as equal to be 0 for RefPicList0[ i ].~~

**chroma\_weight\_l0\_flag**[ i ] equal to 1 specifies that weighting factors for the chroma prediction values of list 0 prediction using RefPicList0[ i ] are present. chroma\_weight\_l0\_flag[ i ] equal to 0 specifies that these weighting factors are not present.

**delta\_chroma\_weight\_l0[** i **][** j **]** is the difference of the weighting factor applied to the chroma prediction values for list 0 prediction using RefPicList0[ i ] with j equal to 0 for Cb and j equal to 1 for Cr.

The variable ChromaWeightL0[ i ][ j ] is specified ~~by ( 1 << ChromaLog2WeightDenom ) + delta\_chroma\_weight\_l0[ i ][ j ]~~ by invoking the process in 8.3.7. ~~When chroma\_weight\_l0\_flag is equal to 1, the~~ The value of ChromaWeightL0[ i ][ j ] shall be in the range of −128 to 127, inclusive. ~~When chroma\_weight\_l0\_flag is equal to 0~~**~~,~~** ~~ChromaWeightL0[ i ][ j ] is inferred to be equal to 2~~~~ChromaLog2WeightDenom~~ ~~for RefPicList0[ i ].~~

**delta\_chroma\_offset\_l0[** i **][** j **]** is the difference of the additive offset applied to the chroma prediction values for list 0 prediction using RefPicList0[ i ] with j equal to 0 for Cb and j equal to 1 for Cr.

The variable ChromaOffsetL0[ i ][ j ] is specified by invoking the process in 8.3.7. ~~as follows:~~

~~shift = 1 << ( BitDepth~~~~C~~ ~~− 1 )~~

~~ChromaOffsetL0[ i ][ j ] = (delta\_chroma\_offset\_l0[i][j] –   
 ( (shift\*ChromaWeightL0[ i ][ j ]) >> ChromaLog2WeightDenom ) − shift ) (7‑70)~~

The variable ChromaOffsetL0[ i ][ j ] shall be in the range of −127 to 128, inclusive. ~~When chroma\_weight\_l0\_flag is equal to 0~~**~~,~~** ~~ChromaOffsetL0[ i ][ j ] is inferred to be equal to 0 for RefPicList0[ i ].~~

**weights\_l1\_present\_flag** equal to 1 specifies that syntax elements used to derive the weighting factors LumaWeightL1[ i ] and ChromaWeightL1 [ i ][ j ] and the additive offsets LumaOffsetL1[ i ] and ChromaOffsetL1 [ i ][ j ] for the luma and chroma prediction values of list 1 prediction are present. When weights\_l1\_present\_flag is equal to 0, syntax elements used to derive the weighting factors and the additive offsets for luma and chroma prediction values of list 1 prediction are not present. When weights\_l1\_present\_flag is not present, it is inferred to be 1.

**delta\_params\_present\_flag**[i] equal to 1 specifies that the syntax elements luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, delta\_chroma\_offset\_l1 are present. When delta\_params\_present\_flag is equal to 0, the syntax elements luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, delta\_chroma\_offset\_l1 are not present.

**luma\_weight\_l1\_flag, delta\_luma\_weight\_l1**, **delta\_luma\_offset\_l1**, **chroma\_weight\_l1\_flag**, **delta\_chroma\_weight\_l1**, **delta\_chroma\_offset\_l1** have the same semantics as luma\_weight\_l0\_flag, delta\_luma\_weight\_l0, delta\_luma\_offset\_l0, chroma\_weight\_l0\_flag, delta\_chroma\_weight\_l0, delta\_chroma\_offset\_l0, respectively, with l0, list 0, and List0 replaced by l1, list 1, and List1, respectively.

The values of LumaWeightL1[ i ], LumaOffsetL1[ i ] , ChromaWeightL1[ i ][ j ], ChromaOffsetL1[ i ][ j ] are derived by invoking the process in 8.3.7. The value of LumaWeightL1[ i ], LumaOffsetL1[ i ], ChromaWeightL1[ i ][ j ], ChromaOffsetL1[ i ][ j ] shall be in the range of −128 to 127, inclusive.

**~~luma\_weight\_lc\_flag, delta\_luma\_weight\_lc~~**~~,~~ **~~luma\_offset\_lc~~**~~,~~ **~~chroma\_weight\_lc\_flag~~**~~,~~ **~~delta\_chroma\_weight\_lc~~**~~,~~ **~~delta\_chroma\_offset\_lc~~** ~~have the same semantics as luma\_weight\_l0\_flag, delta\_luma\_weight\_l0, luma\_offset\_l0, chroma\_weight\_l0\_flag, delta\_chroma\_weight\_l0, delta\_chroma\_offset\_l0, respectively, with l0, list 0, and List0 replaced by lc, list combination, and List combination, respectively.~~

WD changes, including syntax, semantics and decoding process changes, related to the proposed scheme can be found in a separate file attached with this contribution.

# Simulation results

The proposed scheme is implemented on HM6.1. Table 4 and Table 5 compare overhead bits used by pred\_weight\_table() for the current method in [1] and the proposed method for random access main and low delay main settings, respectively. QP = 22 is used. Other settings are the same as in [2] except that in LD setting, ref\_pic\_list\_combination\_flag is set to 0 since L0 and L1 are identical (in RA setting, ref\_pic\_list\_combination\_flag is still set to 1). The fading sequences used are generated using the fading tool in [3]. The average bit overhead comparison over all sequences for RA and LD settings is summarized in Table 6. On average, the proposed scheme increases the bit overhead for pred\_weight\_table() by 2.7% (or 1.9 bits per frame) in RA setting and reduces the bit overhead for pred\_weight\_table() by 48.2% (or 94 bits per frame) in LD setting. Table 4. Weighted prediction table pred\_weight\_table() bits usage comparison (RA main, ref\_pic\_list\_combination\_flag = 1)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fading Sequence | | HM6.1 | proposed | ∆bits/frame | % change |
| FB | Traffic | 4273 | 4403 | 2.0 | 3.04% |
| PeopleOnStreet | 4232 | 4362 | 2.0 | 3.07% |
| NebutaFestival\_10bit | 5151 | 5261 | 0.9 | 2.14% |
| SteamLocomotiveTrain\_10bit | 5474 | 5608 | 1.0 | 2.45% |
| Kimono | 3125 | 3215 | 1.8 | 2.88% |
| ParkScene | 3261 | 3355 | 1.9 | 2.88% |
| Cactus | 6160 | 6358 | 2.0 | 3.21% |
| BasketballDrive | 6123 | 6309 | 1.9 | 3.04% |
| BQTerrace | 7486 | 7756 | 2.1 | 3.61% |
| BasketballDrill | 6615 | 6813 | 2.0 | 2.99% |
| BQMall | 7558 | 7812 | 2.0 | 3.36% |
| PartyScene | 6134 | 6318 | 1.9 | 3.00% |
| RaceHorsesC | 3261 | 3351 | 1.4 | 2.76% |
| BasketballPass | 6674 | 6876 | 2.1 | 3.03% |
| BQSquare | 7732 | 8002 | 2.1 | 3.49% |
| BlowingBubbles | 6108 | 6288 | 1.9 | 2.95% |
| RaceHorses | 3385 | 3479 | 1.4 | 2.78% |
| BasketballDrillText | 6321 | 6505 | 1.9 | 2.91% |
| ChinaSpeed | 4133 | 4255 | 1.9 | 2.95% |
| SlideEditing | 4349 | 4479 | 2.0 | 2.99% |
| SlideShow | 2286 | 2344 | 1.8 | 2.54% |
| FW | Traffic | 5287 | 5417 | 2.0 | 2.46% |
| PeopleOnStreet | 5226 | 5356 | 2.0 | 2.49% |
| NebutaFestival\_10bit | 7126 | 7268 | 1.1 | 1.99% |
| SteamLocomotiveTrain\_10bit | 9374 | 9648 | 2.1 | 2.92% |
| Kimono | 4033 | 4127 | 1.9 | 2.33% |
| ParkScene | 4059 | 4153 | 1.9 | 2.32% |
| Cactus | 7586 | 7788 | 2.1 | 2.66% |
| BasketballDrive | 7157 | 7341 | 1.9 | 2.57% |
| BQTerrace | 8883 | 9141 | 2.0 | 2.90% |
| BasketballDrill | 7825 | 8027 | 2.1 | 2.58% |
| BQMall | 9209 | 9477 | 2.1 | 2.91% |
| PartyScene | 7538 | 7738 | 2.1 | 2.65% |
| RaceHorsesC | 5199 | 5327 | 2.0 | 2.46% |
| BasketballPass | 7813 | 8015 | 2.1 | 2.59% |
| BQSquare | 9262 | 9520 | 2.0 | 2.79% |
| BlowingBubbles | 7675 | 7875 | 2.1 | 2.61% |
| RaceHorses | 5252 | 5382 | 2.0 | 2.48% |
| BasketballDrillText | 7727 | 7929 | 2.1 | 2.61% |
| ChinaSpeed | 5189 | 5319 | 2.0 | 2.51% |
| SlideEditing | 5282 | 5412 | 2.0 | 2.46% |
| SlideShow | 2787 | 2839 | 1.6 | 1.87% |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table 5. Weighted prediction table bits usage comparison (LD main, ref\_pic\_combination\_flag = 0)Fading Sequence | | HM6.1 | proposed | ∆bits/frame | % change |
| FB | Kimono | 8819 | 4576 | -86.6 | -48.11% |
| ParkScene | 9329 | 4831 | -91.8 | -48.22% |
| Cactus | 17091 | 8880 | -84.6 | -48.04% |
| BasketballDrive | 17161 | 8915 | -85.0 | -48.05% |
| BQTerrace | 20563 | 10728 | -76.2 | -47.83% |
| BasketballDrill | 18485 | 9577 | -91.8 | -48.19% |
| BQMall | 21027 | 10960 | -78.0 | -47.88% |
| PartyScene | 17305 | 8987 | -85.8 | -48.07% |
| RaceHorsesC | 9362 | 4890 | -68.8 | -47.77% |
| BasketballPass | 18543 | 9606 | -92.1 | -48.20% |
| BQSquare | 21335 | 11114 | -79.2 | -47.91% |
| BlowingBubbles | 17263 | 8966 | -85.5 | -48.06% |
| RaceHorses | 10212 | 5321 | -75.2 | -47.89% |
| FourPeople | 21155 | 11024 | -78.5 | -47.89% |
| Johnny | 21205 | 11049 | -78.7 | -47.89% |
| KristenAndSara | 20953 | 10923 | -77.8 | -47.87% |
| BasketballDrillText | 17673 | 9171 | -87.6 | -48.11% |
| ChinaSpeed | 11897 | 6171 | -88.1 | -48.13% |
| SlideEditing | 12171 | 6308 | -90.2 | -48.17% |
| SlideShow | 6855 | 3537 | -100.5 | -48.40% |
| FW | Kimono | 11335 | 5834 | -112.3 | -48.53% |
| ParkScene | 11431 | 5882 | -113.2 | -48.54% |
| Cactus | 20787 | 10728 | -103.7 | -48.39% |
| BasketballDrive | 19961 | 10315 | -99.4 | -48.32% |
| BQTerrace | 24279 | 12586 | -90.6 | -48.16% |
| BasketballDrill | 21489 | 11079 | -107.3 | -48.44% |
| BQMall | 25389 | 13141 | -94.9 | -48.24% |
| PartyScene | 20571 | 10620 | -102.6 | -48.37% |
| RaceHorsesC | 14465 | 7455 | -107.8 | -48.46% |
| BasketballPass | 21311 | 10990 | -106.4 | -48.43% |
| BQSquare | 25315 | 13104 | -94.7 | -48.24% |
| BlowingBubbles | 21161 | 10915 | -105.6 | -48.42% |
| RaceHorses | 14545 | 7495 | -108.5 | -48.47% |
| FourPeople | 26265 | 13579 | -98.3 | -48.30% |
| Johnny | 26297 | 13595 | -98.5 | -48.30% |
| KristenAndSara | 25951 | 13422 | -97.1 | -48.28% |
| BasketballDrillText | 21161 | 10915 | -105.6 | -48.42% |
| ChinaSpeed | 14437 | 7441 | -107.6 | -48.46% |
| SlideEditing | 14699 | 7572 | -109.6 | -48.49% |
| SlideShow | 8097 | 4159 | -119.3 | -48.64% |

Table 6. Average bits/frame for pred\_weight\_table() over all sequences for RA and LD settings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| pred\_weight\_table() | HM6.1 | proposed | Difference | % change |
| RA-main | 69.05 | 70.93 | 1.88 | 2.73% |
| LD-main | 195.14 | 101.00 | -94.14 | -48.24% |

As previously mentioned, since the current WP parameter signaling scheme in [1] relies on LC to carry the WP parameters when ref\_pic\_list\_combination\_flag is set to 1, in order to assign different WP parameters to an L0 entry and an L1 entry that represent the same reference picture in the DPB, reference picture list reordering must be performed on LC. This incurs additional bit overhead due to ref\_pic\_list\_combination() as explicit signaling of LC entries must be used. To compare the bit overhead in this case, a simple encoder-only scheme is implemented in HM6.1 to assign different WP parameters to corresponding L0 and L1 entries when they represent the same picture. Since the WP parameters (w, o) have fixed precision, they must be quantized. The encoder examines, during parameter quantization, whether rounding up or rounding down is performed. If rounding up is performed to get WP parameters for L0 entry, then rounding down is performed to get WP parameters for the same picture on L1. And vice versa. With this encoder scheme, simualtion was carried out to measure the overall bit overhead due to 1) pred\_weight\_table() signaling and 2) ref\_pic\_list\_combination() signaling. Random access setting with QP=22 is used. Table 7 shows that overall 17% bit overhead saving (averaged over all sequences) could be achieved using the proposed scheme, which includes savings on both pred\_weight\_table() and ref\_pic\_list\_combination().

Table 7. Weighted prediction table bits usage with different WP parameters on L0 and L1 (RA mode)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fading Sequence | | HM6.1 | | | Proposed | | | % change |
| WP bits | LC bits | Total | WP bits | LC bits | Total |
| FB | Traffic | 5440 | 690 | 6130 | 5042 | 148 | 5190 | -18.11 |
| PeopleOnStreet | 5422 | 690 | 6112 | 5009 | 148 | 5157 | -18.52 |
| NebutaFestival\_10bit | 6823 | 1466 | 8289 | 6066 | 308 | 6374 | -30.04 |
| SteamLocomotiveTrain\_10bit | 7317 | 1466 | 8783 | 6506 | 308 | 6814 | -28.90 |
| Kimono | 3905 | 496 | 4401 | 3609 | 108 | 3717 | -18.40 |
| ParkScene | 4127 | 496 | 4623 | 3809 | 108 | 3917 | -18.02 |
| Cactus | 7970 | 1078 | 9048 | 7394 | 228 | 7622 | -18.71 |
| BasketballDrive | 7914 | 1078 | 8992 | 7246 | 228 | 7474 | -20.31 |
| BQTerrace | 9734 | 1466 | 11200 | 9152 | 308 | 9460 | -18.39 |
| BasketballDrill | 8506 | 1078 | 9584 | 7768 | 228 | 7996 | -19.86 |
| BQMall | 9860 | 1466 | 11326 | 9083 | 308 | 9391 | -20.60 |
| PartyScene | 7896 | 1078 | 8974 | 7289 | 228 | 7517 | -19.38 |
| RaceHorsesC | 4149 | 690 | 4839 | 3751 | 148 | 3899 | -24.11 |
| BasketballPass | 8614 | 1078 | 9692 | 7841 | 228 | 8069 | -20.11 |
| BQSquare | 10092 | 1466 | 11558 | 9384 | 308 | 9692 | -19.25 |
| BlowingBubbles | 7869 | 1078 | 8947 | 7218 | 228 | 7446 | -20.16 |
| RaceHorses | 4381 | 690 | 5071 | 3976 | 148 | 4124 | -22.96 |
| BasketballDrillText | 8198 | 1078 | 9276 | 7515 | 228 | 7743 | -19.80 |
| ChinaSpeed | 5249 | 690 | 5939 | 4823 | 148 | 4971 | -19.47 |
| SlideEditing | 5539 | 690 | 6229 | 5141 | 148 | 5289 | -17.77 |
| SlideShow | 2787 | 302 | 3089 | 2582 | 68 | 2650 | -16.57 |
| FW | Traffic | 6718 | 690 | 7408 | 6084 | 148 | 6232 | -18.09 |
| PeopleOnStreet | 6686 | 690 | 7376 | 6013 | 148 | 6161 | -18.69 |
| NebutaFestival\_10bit | 9461 | 1466 | 10927 | 8228 | 308 | 8536 | -18.53 |
| SteamLocomotiveTrain\_10bit | 12185 | 1466 | 13651 | 11057 | 308 | 11365 | -17.72 |
| Kimono | 5064 | 496 | 5560 | 4584 | 108 | 4692 | -17.71 |
| ParkScene | 5119 | 496 | 5615 | 4617 | 108 | 4725 | -18.16 |
| Cactus | 9821 | 1078 | 10899 | 8828 | 228 | 9056 | -19.00 |
| BasketballDrive | 9249 | 1078 | 10327 | 8309 | 228 | 8537 | -18.45 |
| BQTerrace | 11580 | 1466 | 13046 | 10517 | 308 | 10825 | -17.22 |
| BasketballDrill | 10062 | 1078 | 11140 | 9058 | 228 | 9286 | -19.11 |
| BQMall | 11954 | 1466 | 13420 | 10856 | 308 | 11164 | -17.49 |
| PartyScene | 9745 | 1078 | 10823 | 8770 | 228 | 8998 | -18.81 |
| RaceHorsesC | 6610 | 690 | 7300 | 5995 | 148 | 6143 | -17.80 |
| BasketballPass | 10109 | 1078 | 11187 | 9024 | 228 | 9252 | -19.95 |
| BQSquare | 12081 | 1466 | 13547 | 10901 | 308 | 11209 | -18.12 |
| BlowingBubbles | 9908 | 1078 | 10986 | 8896 | 228 | 9124 | -19.20 |
| RaceHorses | 6686 | 690 | 7376 | 6020 | 148 | 6168 | -18.58 |
| BasketballDrillText | 9952 | 1078 | 11030 | 8989 | 228 | 9217 | -18.69 |
| ChinaSpeed | 6633 | 690 | 7323 | 5991 | 148 | 6139 | -18.22 |
| SlideEditing | 6727 | 690 | 7417 | 6096 | 148 | 6244 | -18.05 |
| SlideShow | 3340 | 302 | 3642 | 3055 | 68 | 3123 | -15.73 |
|  | **Average (bits/frame)** | 88.54 | 10.84 | 99.38 | 80.52 | 2.31 | 82.83 | -16.56 |

# References

1. [B. Bross](mailto:benjamin.bross@hhi.fraunhofer.de), [W.-J. Han](mailto:wjhan.han@samsung.com), [J.-R. Ohm](mailto:ohm@ient.rwth-aachen.de), [G. J. Sullivan](mailto:garysull@microsoft.com), [T. Wiegand](mailto:thomas.wiegand@hhi.fraunhofer.de). WD6: Working Draft 6 of High-Efficiency Video Coding. Document no JCTVC-H1003. February 2012.
2. F. Bossen. Common HM test conditions and software reference configurations. Document no JCTVC-G1200. November 2011.
3. P. Bordes, T.K. Tan, “JCT-VC AHG report: Weighted prediction (AHG 18)”, Document no JCTVC-F018, July 2011.
4. A. Tanizawa, T. Chujoh, T. Yamakage, Redundancy removal of explicit weighted prediction syntax. Document no JCTVC-G441, Nov 2011.
5. Y. Suzuki, et al, Extension of uni-prediction simplification in B slices. Document no JCTVC-D421, January 2011.
6. Y. He and Y. Ye, AHG21: Unification of reference picture list modification processes. Document no JCTVC-H0138, Feb 2012.

# Patent rights declaration(s)

**InterDigital Communications, LLC may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**