|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  11th Meeting: Shanghai, CN, 10–19 Oct. 2012 | Document: JCTVC-K0239 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Temporal motion vector prediction hook for MV-HEVC** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Ying Chen Li Zhang Vadim Seregin Marta Karczewicz  5775 Morehouse Dr  San Diego, CA 92121 USA | Tel: Email: | 1-858-845-6589 [cheny@qti.qualcomm.com](mailto:cheny@qti.qualcomm.com)  1-858-651-6660 [lizhang@qti.qualcomm.com](mailto:lizhang@qti.qualcomm.com)  1-858-651-3481 [vseregin@qti.qualcomm.com](mailto:vseregin@qti.qualcomm.com)  1-858-658-5673 [martak@qti.qualcomm.com](mailto:martak@qti.qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

In the current HEVC design, when merge mode is used, the reference index for the temporal merging candidate is always zero. This impacts the coding efficiency relatively small, since scaling can be used to compensate the temporal location differences. However, in the context of multiview or 3DV coding, reference index equal to zero may correspond to the reference picture in the same view, while the motion vector (MV) of the co-located PU may point to an inter-view reference picture. In this case, TMVP candidate is considered as unavailable. To address this issue, it is proposed that one additional target reference index is used, so that TMVP can be supported even when the MV of the co-located PU points to an inter-view reference picture. For multiview video coding (MV-HEVC), the proposed method provides about 0.94% average bitrate saving for the all the views and 2.5% bitrate saving for the non-base views.

# Introduction

## Multiview video coding

MV-HEVC is the multiview extension of HEVC to support multiview video coding and it is being developed under JCT-3V. Similar to MVC (the multiview extension of AVC), MV-HEVC doesn’t contain any low-level changes beyond HEVC. In MV-HEVC, an inter-view reference picture is marked as long-term.

In MV-HEVC, one view can be predicted from the other by the so-called disparity motion compensation. It is realized in a way that a decoded view component from a different view but in the same time instance can be added in a reference picture list of the current view component. For example, as shown in Figure 1, wherein the vertical indices (V1, V0 and V2) of the pictures correspond to view identifiers and the horizontal indices (T0 through T11) correspond to POC values. Each square is a view component and when a vertical arrow links view component A (e.g., T5/V0) to another view component B (e.g., T5/V1, in the same time instance), view component A can be added into the reference picture list 0 of view component B.

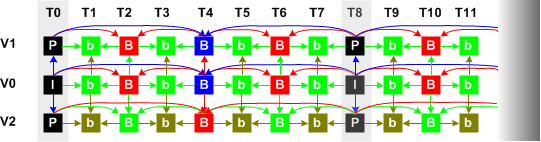


Figure 1: Typical motion prediction structure in 3DV (texture only).

Note that in the above example, both view component A and view component B have the same POC value (5).

A motion vector pointing to a view component of a different view is called disparity motion vector in this document and this view component is referred to as an inter-view reference picture.

In MV-HEVC, an inter-view reference picture is marked as long-term.

## Merge mode in HEVC/MV-HEVC

In HEVC, when the current mode is merge, the target reference index is always set to 0. The motion vector in the co-located PU, if referring to a short-term (ST) reference picture, is scaled to form a merge candidate of the current PU (PU0), as shown in Figure 2, wherein MV0 is scaled to MV0’ during the merge mode.

However, if the co-located PU has a motion vector (MV1) referring to an inter-view reference picture, marked as long-term, the motion vector is not used to predict the current PU (PU1).

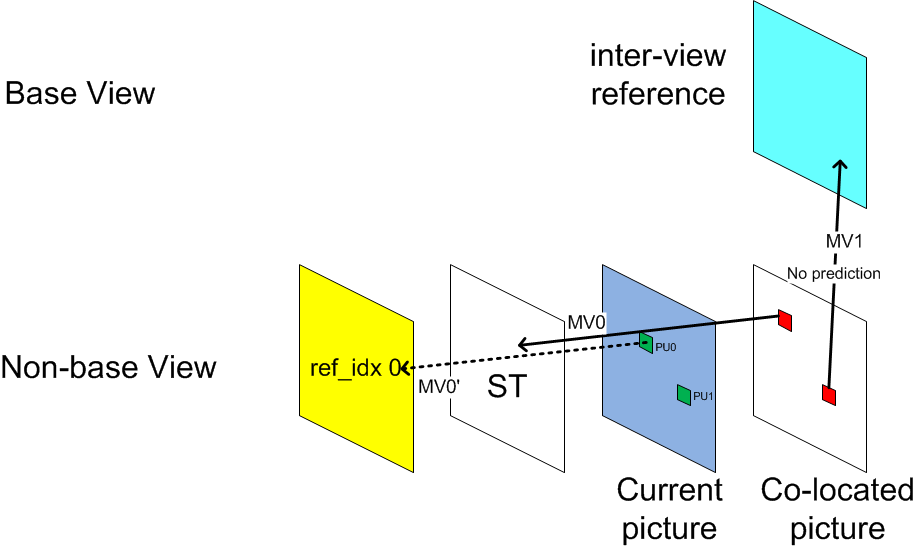


Figure 2: TMVP in MV-HEVC (merge mode).

In MV-HEVC, the temporal motion vector prediction process is the same as in HEVC, due to the fact that no modifications in the coding unit or lower level can be introduced.

However, there might be significant amount of co-located PUs (in the co-located picture) which contain motion vectors referring to an inter-view reference picture while the target reference index (being equal to 0) indicates a short-term reference picture. Therefore, disabling prediction from those motion vectors makes the merge mode less efficient.

# Proposal

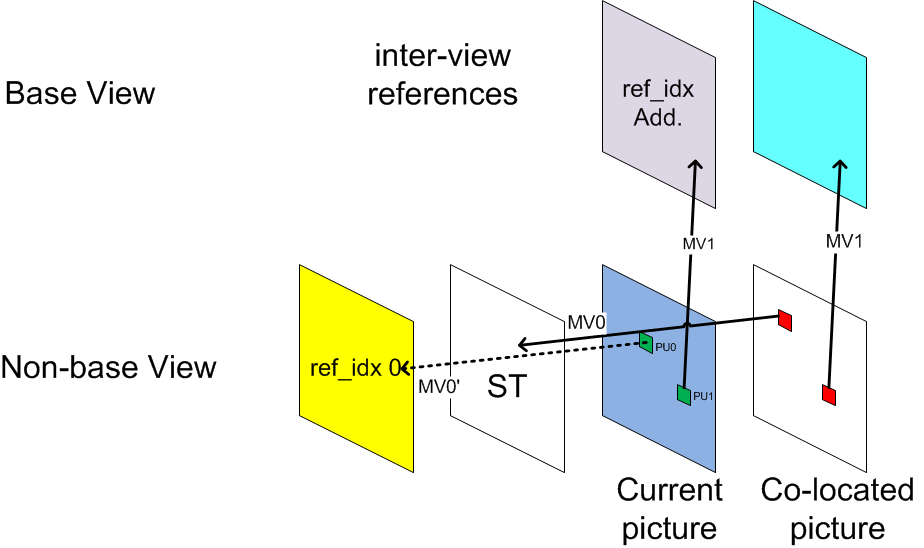


Figure 3: Proposed TMVP in MV-HEVC (merge mode).

It is proposed that an additional target reference index is enabled. In the above example, all short-term reference pictures are still scaled towards the reference picture with reference index equal to 0, as in the current HEVC specification. However, for all long-term reference pictures, a new additional reference index (ref\_idx Add.) is derived, so that the motion vectors referring to a long-term reference picture can be used to form a merge candidate and not considered as unavailable. As shown in Figure 3, MV1 of the co-located block of PU1 can be used to form a disparity motion vector candidate.

In general, when the reference index equal to 0 represents a short-term reference picture, the newly added target reference index is used to represent a long-term reference picture. When the reference index equal to 0 represents a long-term reference picture, the newly added target reference index is used to represent a short-term reference picture.

Note that similar to the current HEVC specification, if the target reference index corresponds to a long-term reference picture, the motion vector in a co-located block, if considered as available, is not scaled.

## Derivation of the additional target reference index

In the slice level, the value of the additional target reference index (refIdxLXA) for each of RefPicList0 and RefPicList1, if available, is derived.

**8.3.5 Derivation process for the additional target reference index for TMVP**

This process is invoked when the current slice is a P or B slice. Additional target reference indices refIdxL0A and refIdxL1A are derived.

Set variables refIdxL0A and refIdxL1A both to -1.

The following apply to derive refIdxL0A.

bZeroIdxLongTermFlag = RefPicList0[ 0 ] is a short-term reference picture ? 0 : 1  
bFound = 0  
for( i = 1; i <= num\_ref\_idx\_l0\_active\_minus1&&! bFound; i++)  
 if ( (bZeroIdxLongTermFlag && RefPicList0[ i ] is a short-term reference picture) | |  
 (!bZeroIdxLongTermFlag && RefPicList0[ i ] is a long-term reference picture) ) {  
 refIdxL0A = i  
 bFound =1  
 }

When the slice is a B slice, the following apply to derive refIdxL1A.

bZeroIdxLongTermFlag = RefPicList1[ 0 ] is a short-term reference picture ? 0 : 1  
bFound = 0  
for( i = 1; i <= num\_ref\_idx\_l1\_active\_minus1&&! bFound; i++)  
 if ( (bZeroIdxLongTermFlag && RefPicList1[ i ] is a short-term reference picture) | |  
 (!bZeroIdxLongTermFlag && RefPicList1[ i ] is a long-term reference picture) ) {  
 refIdxL1A = i  
 bFound =1  
 }

## Temporal motion vector prediction

During temporal motion vector predition, when the current mode is merge, the taregt reference index 0 may be changed to refIdxLXA (with X being equal to 0 or 1). The AMVP mode is not changed.

The proposed specification text changes are as follows, with newly added/modified text highlighted in green and deleted text marked as red strikethrough ~~(red strikethrough)~~.

### Changes for the invocation of TMVP for merge mode

**8.5.2.1.1 Derivation process for luma motion vectors for merge mode**

This process is only invoked when PredMode[ xC ][ yC ] is equal to MODE\_SKIP or PredMode[ xC ][ yC ] is equal to MODE\_INTER and merge\_flag [ xP ][ yP ] is equal to 1, where ( xP, yP ) specify the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture.

Inputs of this process are

* a luma location ( xC, yC ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a luma location ( xP, yP ) of the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* a variable nCS specifying the size of the current luma coding block,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are

– the luma motion vectors mvL0 and mvL1,

– the reference indices refIdxL0 and refIdxL1,

– the prediction list utilization flags predFlagL0 and predFlagL1.

The variables singleMCLFlag is derived as follows.

* If log2\_parallel\_merge\_level\_minus2 is greater than 0 and nCS is equal to 8, singleMCLFlag is set to 1.
* Otherwise, singleMCLFlag is set to 0.

When singleMCLFlag is equal to 1, xP is set equal to xC, yP is set equal to yC, and both nPbW and nPbH are set equal to nCS.

NOTE – When singleMCLFlag is equal to 1, all the prediction units of the current coding unit share a single merge candidate list, which is identical to the merge candidate list of the 2Nx2N prediction unit.

The motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, and the prediction utilization flags predFlagL0 and predFlagL1 are derived as specified by the following ordered steps:

1. The derivation process for merging candidates from neighboring prediction unit partitions in subclause 8.5.2.1.2 is invoked with the luma coding block location ( xC, yC ), the coding block size nCS, the luma prediction block location ( xP, yP ), the variable singleMCLFlag, the width and the height of the luma prediction block nPbW and nPbH and the partition index partIdx as inputs and the output is assigned to the availability flags availableFlagN, the reference indices refIdxL0N and refIdxL1N, the prediction list utilization flags predFlagL0N and predFlagL1N and the motion vectors mvL0N and mvL1N with N being replaced by A0, A1, B0, B1 or B2.
2. The reference index for temporal merging candidate refIdxLX (with X being 0 or 1) is set equal to 0.
3. The derivation process for temporal luma motion vector prediction in subclause 8.5.3.1.7 is invoked with luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, ~~and~~ refIdxLX and mergeTMVP equal to 1 as the inputs and with the output being the availability flag availableFlagLXCol and the temporal motion vector mvLXCol. The variables availableFlagCol and predFlagLXCol (with X being 0 or 1, respectively) are derived as specified below.
4. **…**

**…**

### Changes for the invocation of TMVP for AMVP mode

**8.5.3.1.5 Derivation process for luma motion vector prediction**

Inputs to this process are

* a luma location ( xC, yC ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a variable nCS specifying the size of the current luma coding block,
* a luma location ( xP, yP ) specifying the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* the reference index of the current prediction unit partition refIdxLX (with X being 0 or 1),
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Output of this process is

* the prediction mvpLX of the motion vector mvLX (with X being 0 or 1).

The motion vector predictor mvpLX is derived in the following ordered steps.

1. The derivation process for motion vector predictor candidates from neighboring prediction unit partitions in subclause 8.5.3.1.6 is invoked with the luma coding block location ( xC, yC ), the coding block size nCS, the luma prediction block location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, refIdxLX (with X being 0 or 1, respectively), and the partition index partIdx as inputs and the availability flags availableFlagLXN and the motion vectors mvLXN with N being replaced by A, B as the output.
2. If both availableFlagLXA and availableFlagLXB are equal to 1 and mvLXA is not equal to mvLXB, availableFlagLXCol is set equal to 0, otherwise, the derivation process for temporal luma motion vector prediction in subclause 8.5.3.1.7 is invoked with luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, ~~and~~ refIdxLX (with X being 0 or 1, respectively) and mergeTMVP equal to 0 as the inputs and with the output being the availability flag availableFlagLXCol and the temporal motion vector predictor mvLXCol.

**…**

### Changes for TMVP

**8.5.3.1.7 Derivation process for temporal luma motion vector prediction**

Inputs to this process are

* a luma location ( xP, yP ) specifying the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* a reference index refIdxLX (with X being 0 or 1)~~.~~,
* a variable mergeTMVP.

Outputs of this process are

* the motion vector prediction mvLXCol,
* the availability flag availableFlagLXCol.

The function RefPicOrderCnt( picX, refIdx, LX ) returns the picture order count PicOrderCntVal of the reference picture with index refIdx from reference picture list LX of the picture picX and is specified as follows.

RefPicOrderCnt( picX, refIdx, LX ) = PicOrderCnt(RefPicListX[ refIdx ] of the picture picX) (8 141)

Depending on the values of slice\_type, collocated\_from\_l0\_flag, and collocated\_ref\_idx, the variable colPic, specifying the picture that contains the collocated partition, is derived as follows.

* If slice\_type is equal to B and collocated\_from\_l0\_flag is equal to 0, the variable colPic specifies the picture that contains the collocated partition as specified by RefPicList1[ collocated\_ref\_idx ].
* Otherwise (slice\_type is equal to B and collocated\_from\_l0\_flag is equal to 1 or slice\_type is equal to P), the variable colPic specifies the picture that contains the collocated partition as specified by RefPicList0[ collocated\_ref\_idx ].

Variable colPb and its position ( xPCol, yPCol ) are derived in the following ordered steps:

1. The variable colPb is derived as follows

yPRb = yP + nPbH (8‑151)

* + If ( yP >> Log2CtbSizeY ) is equal to ( yPRb >> Log2CtbSizeY ), the horizontal component of the right-bottom luma location of the current luma prediction block is defined by

xPRb = xP + nPbW (8‑152)

and the variable colPb is set as the luma prediction block covering the modified location given by ( ( xPRb >> 4 ) << 4, ( yPRb >> 4 ) << 4 ) inside the colPic.

* + Otherwise ( ( yP >> Log2CtbSizeY ) is not equal to ( yPRb >> Log2CtbSizeY ) ), colPb is marked as "unavailable".

1. When colPb is coded in an intra prediction mode or colPb is marked as "unavailable", the following applies.
   * Central luma location of the current prediction block is defined by

xPCtr = ( xP + ( nPbW >> 1 ) (8‑153)

yPCtr = ( yP + ( nPbH >> 1 ) (8‑154)

* + The variable colPb is set as the luma prediction block covering the modified location given by ( ( xPCtr >> 4 ) << 4, ( yPCtr >> 4 ) << 4 ) inside the colPic.

1. ( xPCol, yPCol ) is set equal to the top-left sample of the colPb relative to the top-left luma sample of the colPic.

refIdxLX is set to be refIdxLXA if all of the following conditions are true.

* mergeTMVP is equal to 1.
* LongTermRefPic( currPic, refIdxLX, ListX ) is not equal to LongTermRefPic( colPic, refIdxCol, listCol ).
* refIdxLXA is larger than 0.

The variables mvLXCol and availableFlagLXCol are derived as follows.

* If one or more of the following conditions are true, both components of mvLXCol are set equal to 0 and availableFlagLXCol is set equal to 0.
  + colPb is coded in an intra prediction mode.
  + colPb is marked as "unavailable".
  + slice\_temporal\_mvp\_enable\_flag is equal to 0.
  + LongTermRefPic( currPic, refIdxLX, ListX ) is not equal to LongTermRefPic( colPic, refIdxCol, listCol ).
* Otherwise, the motion vector mvCol, the reference index refIdxCol, and the reference list identifier listCol are derived as follows.
  + If PredFlagL0[ xPCol ][ yPCol ] is equal to 0, mvCol, refIdxCol, and listCol are set equal to MvL1[ xPCol ][ yPCol ], RefIdxL1[ xPCol ][ yPCol ], and L1, respectively.
  + Otherwise (PredFlagL0[ xPCol ][ yPCol ] is equal to 1), the following applies.
  + If PredFlagL1[ xPCol ][ yPCol ] is equal to 0, mvCol, refIdxCol, and listCol are set equal to MvL0[ xPCol ][ yPCol ], RefIdxL0[ xPCol ][ yPCol ], and L0, respectively.
  + Otherwise (PredFlagL1[ xPCol ][ yPCol ] is equal to 1), the following assignments are made.
    - * If PicOrderCnt( pic ) of every picture pic in every reference picture lists is less than or equal to PicOrderCntVal, mvCol, refIdxCol, and listCol are set equal to MvLX[ xPCol ][ yPCol ], RefIdxLX[ xPCol ][ yPCol ] and LX, respectively with X being the value of X this process is invoked for.
      * Otherwise (PicOrderCnt( pic ) of at least one picture pic in at least one reference picture list is greater than PicOrderCntVal, mvCol, refIdxCol and listCol are set equal to MvLN[ xPCol ][ yPCol ], RefIdxLN[ xPCol ][ yPCol ] and LN, respectively with N being the value of collocated\_from\_l0\_flag.

and the variable availableFlagLXCol is set equal to 1 and the following applies.

* + If RefPicListX[ refIdxLX ] is a long-term reference picture, or PicOrderCnt( colPic ) – RefPicOrderCnt( colPic, refIdxCol, listCol ) is equal to PicOrderCntVal – PicOrderCnt( RefPicListX[ refIdxLX ] ),

mvLXCol = mvCol (8‑155)

* + Otherwise, mvLXCol is derived as scaled version of the motion vector mvCol as specified below

tx = ( 16384 + ( Abs( td ) >>1 ) ) / td (8‑156)

distScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (8‑157)

mvLXCol =  Clip3( −32768, 32767, Sign2( distScaleFactor \* mvCol ) \*    
 ( (Abs( distScaleFactor \* mvCol ) + 127 ) >> 8 ) ) (8‑158)

where td and tb are derived as

td = Clip3( −128, 127, PicOrderCnt( colPic ) – RefPicOrderCnt( colPic, refIdxCol, listCol ) ) (8‑159)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListX [ refIdxLX ] ) ) (8‑160)

# Simulation results

This section provides simulation results of the proposed method based on the 3DV-HTM software, under the JCT-3V common test conditions for coding texture views.

Since the latest 3DV HTM software (HTM4.0) is based on HM6, changes are made to the anchor such that the target reference index for the temporal merging candidate is always 0 as defined in the latest HEVC specification. Besides, all low-level coding tools such as inter-view motion prediction and inter-view residual prediction are disabled.

The coding gain of the proposed method compared to the anchor for texture views coding in 3-veiw case is reported in this section. Table 1 shows the average coding gain of the proposed method with respect to HTM4.0 anchor. The column denoted by ‘Video only’ lists the coding gain of all three texture views where the bitrates represent the total bitrates of bitstreams containing three texture views and the PSNR values are the average PSNR values of the three decoded texture views. Bitrate savings for “Video 1” and “Video 2” are also listed in this table. Note that “Video 0” corresponds to the base view, while “Video 1” and “Video 2” indicate the non-base views (texture only).

As shown in Table 1, the overall average bitrate saving is around 0.94% for all texture views and the saving of the non-base views is about 2.5%.

Table 1: Coding gain with respect to HTM4.0 anchor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Video 0 | Video 1 | Video 2 | Video only |
| Balloons | 0.0% | -3.13% | -2.94% | -1.39% |
| Kendo | 0.0% | -2.42% | -2.28% | -1.03% |
| Newspapercc | 0.0% | -1.55% | -1.17% | -0.59% |
| GhostTownFly | 0.0% | -5.62% | -5.38% | -1.54% |
| PoznanHall2 | 0.0% | -1.61% | -1.24% | -0.64% |
| PoznanStreet | 0.0% | -2.12% | -2.01% | -0.71% |
| UndoDancer | 0.0% | -1.86% | -1.90% | -0.69% |
| 1024x768 | 0.0% | -2.37% | -2.13% | -1.00% |
| 1920x1088 | 0.0% | -2.80% | -2.63% | -0.90% |
| average | 0.0% | -2.62% | -2.42% | -0.94% |

We also provided an anchor software branch of HTM, to fully align with MV-HEVC, in terms of motion and high-level syntax, such as reference picture marking. The performance of the proposed method in the MV-HEVC software is listed in Table 2. Similar coding gain can be achieved in this MV-HEVC software version.

Table 2: Coding gain with respect to MV-HEVC anchor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Video 0 | Video 1 | Video 2 | Video only |
| Balloons | 0.0% | -3.16% | -2.93% | -1.40% |
| Kendo | 0.0% | -2.50% | -2.22% | -1.03% |
| Newspapercc | 0.0% | -1.52% | -1.20% | -0.59% |
| GhostTownFly | 0.0% | -5.70% | -5.49% | -1.55% |
| PoznanHall2 | 0.0% | -1.92% | -1.03% | -0.67% |
| PoznanStreet | 0.0% | -2.14% | -2.06% | -0.71% |
| UndoDancer | 0.0% | -1.89% | -2.00% | -0.69% |
| 1024x768 | 0.0% | -2.39% | -2.12% | -1.01% |
| 1920x1088 | 0.0% | -2.91% | -2.64% | -0.91% |
| average | 0.0% | -2.69% | -2.42% | -0.95% |

# Patent rights declaration(s)

**Qualcomm Incorporated 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).**