|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  6th Meeting: Torino, IT, 14-22 July, 2011 | Document: JCTVC-F612 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Modifications to intra blocks coefficient coding with VLC** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Marta Karczewicz, Yunfei Zheng, Liwei Guo, Xianglin Wang  5775 Morehouse Dr.  San Diego, CA 92121, USA | Tel:  Email: | 1-858-658-5673 [martak@qualcomm.com](mailto:martak@qualcomm.com)  [zhengy@qualcomm.com](mailto:zhengy@qualcomm.com)  [liweig@qualcomm.com](mailto:liweig@qualcomm.com)  [xianglin@qualcomm.com](mailto:xianglin@qualcomm.com) |
| *Source:* | Qualcomm Inc. | | |

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

# Abstract

Coding results after modifying the tables used in VLC coding are presented. It is further proposed to extend usage of horizontal and vertical scans to 16x16 and 32x32 blocks. In addition this contribution proposes to modify coefficient coding for 16x16 and 32x32 blocks.

# Introduction

Three modifications to VLC coding of intra blocks transform coefficients are proposed:

1. New VLC tables are designed to reflect change in the coefficient statistic due to the introduction of horizontal and vertical scan for 4x4 and 8x8 blocks (shorter runs of zeros). Moreover it is shown that in some cases it is beneficial for 16x16 and 32x32 blocks to have their own VLC tables and do not share them tables with 8x8 blocks.
2. Usage of horizontal and vertical scans is extended to 16x16 and 32x32 blocks.
3. CABAC can take advantage of spatial relationship between coefficients in 16x16 and 32x32 blocks by using context based on neighboring coefficients while coding significant bit and using 4x4 sub-blocks when coding level information. In case of VLC we propose dividing 16x16 and 32x32 transform coefficients blocks into 4x4 sub-blocks and sending information for the sub-block whether it has any non-zero coefficients.

# Proposal

## Table Modifications

It is proposed to modify which VLC table is used to code syntax element run\_level\_one. The new mapping between index of the VLC table, vlcNum, and maximum run maxRunIdx for intra luma 4x4 blocks and intra luma 8x8 blocks (tableIdx=0) and intra luma 16x16 and 32x32 blocks (tableIdx=1) is given in Table 2‑1.

It is further proposed to modify the tables used to derive variable largeOnePos from trOne and maxRunIdx for intra luma blocks. The new derivation is given in Table 2‑2~Table 2‑4.

Table ‑ – Derivation of vlcNum from tableIdx and maxRunIdx

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | | |
| **tableIdx** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| 0 | 8 | 0 | 0 | 0 | 0 | 1 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 1 | 8 | 0 | 0 | 0 | 0 | 1 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | |
| **tableIdx** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** |
| 0 | 5 | 5 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 5 | 5 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 |

Table ‑ – Derivation of largeOnePos from tableIdx and maxRunIdx when N is equal to 4

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | | |
| **trOne** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 2 | 2 | 2 | 2 | 3 | 2 | 2 | 4 | 4 | 2 | 4 | 4 | 2 | 2 | 2 |
| 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | NA |
| 3 | 1 | 1 | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | NA | NA |
| 4 | 1 | 1 | 1 | 2 | 1 | 1 | 2 | 2 | 0 | 0 | 2 | 0 | NA | NA | NA |

Table ‑ – Derivation of largeOnePos from tableIdx and maxRunIdx when N is equal to 8

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | | |
| **trOne** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| 0,4 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 4 | 2 | 2 | 2 | 2 | 2 | 4 | 4 |
| 1-3 | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 8 | 4 | 2 | 4 | 4 | 4 | 6 | 6 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | |
| **trOne** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** |
| 0,4 | 6 | 4 | 4 | 4 | 4 | 4 | 4 | 6 | 6 | 4 | 4 | 4 | 6 | 6 |
| 1-3 | 8 | 6 | 6 | 6 | 8 | 8 | 8 | 10 | 12 | 8 | 8 | 8 | 10 | 14 |

Table ‑ – Derivation of largeOnePos from tableIdx and maxRunIdx when N is greater than 8

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | | |
| **trOne** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** |
| 0,4 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 4 |
| 1-3 | 1 | 2 | 2 | 3 | 4 | 4 | 4 | 6 | 6 | 6 | 6 | 6 | 8 | 14 | 10 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **maxRunIdx** | | | | | | | | | | | | | |
| **trOne** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** |
| 0,4 | 4 | 4 | 6 | 6 | 6 | 4 | 4 | 6 | 6 | 6 | 6 | 8 | 6 | 10 |
| 1-3 | 8 | 8 | 10 | 10 | 16 | 12 | 10 | 10 | 12 | 12 | 12 | 28 | 16 | 22 |

## Horizontal and Vertical Scan for 16x16 and 32x32 Blocks

It is proposed to extend usage of horizontal and vertical scan to 16x16 and 32x32 blocks. Table 2‑5 lists value of index scanIdx (or ScanType) for luma intra prediction modes for 16x16 and 32x32 blocks. Index scanIdx equal to 0 specifies usage of a zig-zag scan, scanIdx equal to 1 specifies a horizontal scan and scanIdx equal to 2 specifies a vertical scan.

Table ‑ – Values of ScanType[ log2TrafoSize ][ IntraPredMode ] for block 16x16 and 32x32

|  |  |  |
| --- | --- | --- |
| **IntraPredMode** | **log2TrafoSize** | |
| **4 (16x16)** | **5 (32x32)** |
| 0 | 1 | 1 |
| 1 | 2 | 2 |
| 2-20 | 0 | 0 |
| 21-22 | 1 | 0 |
| 23-28 | 0 | 0 |
| 29-30 | 2 | 0 |
| 31-35 | 0 | 0 |

## Sub-Block Coding

The 16x16 and 32x32 transform coefficient blocks are divided into 4x4 sub-blocks. After the position of the last nonzero coefficient, lastPos, in 16x16 or 32x32 block is sent, for each 4x4 sub-block at position (xS,yS) two conditions are checked:

1. The number of coded coefficients in this sub-block is larger than 4: noCoeff[xS][yS]>4.
2. The average number of non-zero coefficients in this sub-block of the previously coded blocks is less than a threshold, i.e. noCodedCoeff[xS][yS]<=subBlkThreshold. Values of variable subBlkThreshold are specified in Table 2‑6.

Table ‑ – Derivation of subBlkThreshold from noSubBlks[xS][yS]

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **noSubBlks[xS][yS]** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| subBlkThreshold | 32 | 2 | 4 | 7 | 9 | 12 | 14 | 17 | 19 | 22 | 24 | 27 | 29 | 32 | 32 | 32 |

For each sub-block for which both conditions are true a single bit is sent. The bit is equal to zero if all the transform coefficients in this sub-block are zero. Remaining coefficients in 16x16 or 32x32 block are coded using current coding method for these blocks.

Values in the variable array noCoeff[xS][yS] are initialized to 0 before encoding 16x16 or 32x32 block and calculated using value of lastPos:

for (pos=0; pos<= lastPos; pos++) {

xC= ScanOrder [log2TrafoSize−2][scanIdx][pos][0];

yC= ScanOrder [log2TrafoSize−2][scanIdx][pos][1];

noCoeff[xC/4] [yC/4]++;

}

where the scanning order array ScanOrder specifies the mapping of the scan position pos to transfrom coeffcient position (xC, yC) within the current transform block. Varaible log2TrafoSize has value of 4 for 16x16 and 5 for 32x32 blocks.

After the the transform coeffcient level block, transCoeffLevel, is encoded, the variable arrays noSubBlks and noCodedCoeff are updated as follows:

for (pos=0; pos<= lastPos; pos++){

xC= ScanOrder [log2TrafoSize−2][scanIdx][pos][0];

yC= ScanOrder [log2TrafoSize−2][scanIdx][pos][1];

if (transCoeffLevel[xC][yC] != 0){

noCodedCoeff[log2TrafoSize−4][scanIdx] [xC/4][yC/4]++;

}

}

for (xS=0; xS< blkSize/4; xS++)

for (yS=0; yS< blkSize/4; yS++)

if (noCoeff[xS] [yS]>0)

noSubBlks[log2TrafoSize−4][scanIdx][xS][yS]++;

Values of noSubBlks and noCodedCoeff for sub-lock at position (xS, yS) are divided by 2 if any of them becomes larger than 16.

# Coding results

Simulations are performed using all three low complexity configurations. Detailed results can be found in the associated excel data sheet.

The results show that when modifying intra VLC tables, an average coding gain of 0.48%, 0.19% and 0.06 % can be obtained respectively with all intra, low delay and random access configuration. Extending usage of horizontal and vertical scan to 16x16 and 32x32 blocks brings an average coding gain of 0.36%, 0.21%, and 0.18% for the same configurations. Sub-block coding brings an average coding gain of 0.65%, 0.25%, and 0.09% for these configurations.

When all these 3 modifications are applied the average gain of 1.41%, 0.57%, and 0.28% can be observed for all intra, low delay and random access configurations.

# Patent rights declaration(s)

**Qualcomm 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).**

# Annex: proposed text change with sub-block coding

## Residual coding CAVLC syntax

|  |  |
| --- | --- |
| residual\_coding\_cavlc( x0, y0, log2TrafoSize, trafoDepth, scanIdx, cIdx ) { | Descriptor |
| n = ( 1 << ( log2TrafoSize << 1 ) ) >> ( cIdx > 0  ?  2  :  0 ) |  |
| **last\_pos\_level\_one** | ce(v) |
| lastPos **=** last\_pos\_level\_one % n |  |
| levelGreaterThanOneFlag = ( last\_pos\_level\_one > n ) |  |
| level\_and\_sign( levelGreaterThanOneFlag, transCoeffLevel, x0, y0, trafoDepth,  cIdx, lastPos) |  |
| if (log2TrafoSize >= 4) { |  |
| numBlocks = n>>4 |  |
| getNumCoeffs ( numCoeffs, log2TrafoSize, scanIdx, lastPos) |  |
| for ( i = 0; i < numBlocks; i++) |  |
| zeroBlk [i]= (numCodedCoeff [i] <= subBlkThreshold[noSubBlks] ) ? 1:0 |  |
| zeroBlk[getBlkIndex(log2TrafoSize, scanIdx, lastPos)] = 0 |  |
| for ( i = 0; i < numBlocks; i++) { |  |
| if (numCoeffs [ i ] > 4 && zeroBlk [i] ) { |  |
| **coded\_blk\_flag** | u(1) |
| zeroBlk[i] = coded\_blk\_flag |  |
| } |  |
| } |  |
| numZeroes = 0 |  |
| for ( i = 0; i <= lastPos; i++) |  |
| if( zeroBlk[getBlkIndex(log2TrafoSize, scanIdx, i)] ) |  |
| numZeroes ++ |  |
| lastPos = lastPos – numZeroes |  |
| } |  |
| runModeFlag = TRUE |  |
| n = lastPos – 1 |  |
| sumBigCoeff = 0 |  |
| switchThres = ( PredMode == MODE\_INTRA && cIdx == 0)? 0 : 49 |  |
| while (runModeFlag && n >= 0) { |  |
| **run\_level\_one** | ce(v) |
| runOfZeros = run\_level\_one % n |  |
| levelGreaterThanOneFlag = (run\_level\_one > n) |  |
| trOne = (trOne = = 0 | | levelGreaterThanOneFlag) ? 0 : Max(4, trOne + 1) |  |
| if (log2TrafoSize >= 4) { |  |
| i = 0 |  |
| while( i < runOfZeros ) |  |
| zeroBlk[getBlkIndex(log2TrafoSize, scanIdx, n - i)] ? (i++): (n--) |  |
| } |  |
| n = n – runOfZeros |  |
| if( n >= 0 ) { |  |
| level\_and\_sign( levelGreaterThanOneFlag, transCoeffLevel, x0, y0,  trafoDepth, cIdx,  n) |  |
| if ( levelGreaterThanOneFlag ) { |  |
| sumBigCoeff = sumBigCoeff + level |  |
| if ( log2TrafoSize==2 | | n > switchThres | | sumBigCoeff > 2 ) |  |
| runModeFlag = FALSE |  |
| } |  |
| } |  |
| n-- |  |
| } |  |
| while ( n >= 0 ) { |  |
| **level** | ce(v) |
| if ( level > 0 ) { |  |
| **sign\_flag** | u(1) |
| sign = 1 – 2 \* sign\_flag |  |
| transCoeffLevel[ x0 ][ y0 ][ trafoDepth ][ cIdx ][ n ] = level \* sign |  |
| } |  |
| n-- |  |
| } |  |
| } |  |

*Function definition:*

getBlkIndex ( log2TrafoSize, scanIdx, pos)

{

xC = ScanOrder[ log2TrafoSize − 2 ][ scanIdx  ][ pos ][ 0 ]

yC = ScanOrder[ log2TrafoSize − 2 ][ scanIdx  ][ pos ][ 1 ]

return( (yC>>2<< (log2TrafoSize-2)) + (xC>>2) )

}

getNumCoeffs ( numCoeffs, log2TrafoSize, scanIdx, lastPos)

{

numCoeffs[i] = 0, with i = 0…numBlocks-1

for ( i = 0; i <= lastPos; i++)

numCoeffs[ getBlkIndex(log2TrafoSize, scanIdx, i) ] ++

}