|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  8th Meeting: San José, CA, USA, 1–10 February, 2012 | Document: JCTVC-H0678 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Non-CE11: Signaling of significant\_coeffgroup\_flag for significance map** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Vivienne Sze  Akria Osamoto | Email: | [sze@ti.com](mailto:sze@ti.com) |
| *Source:* | Texas Instruments | | |

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

# Abstract

The significant\_coeffgroup\_flag is used to signal whether a 4x4 sub-block in a 16x16/32x32 TU contains any non-zero significant\_coeff\_flag. In HM-5.0, significant\_coeffgroup\_flag is interleaved with significant\_coeff\_flag. The presence of significant\_coeff\_flag depends on the significant\_coeffgroup\_flag. This contribution proposes signaling all significant\_coeffgroup\_flag in a TU before signaling significant\_coeff\_flag. This reduces the number of context switching and enables earlier context prefetch for significant\_coeff\_flag. Grouping of the significant\_coeffgroup\_flag is expected to have negligible impact on coding efficiency.

# Introduction

The significant\_coeffgroup\_flag is used to signal whether a 4x4 sub-block in a 16x16/32x32 TU contains any non-zero significant\_coeff\_flag. HM-5.0, significant\_coeffgroup\_flags (SCGF) are interleaved with significant\_coeff\_flag (SCF). significant\_coeffgroup\_flags is sent before the significant\_coeff\_flag of the sub-block and then the next significant\_coeffgroup\_flag is sent as shown in Fig. 1. Thus, the context needs to keep switching from SCGF to SCF.

In addition, since the presence of SCF depends on SCGF, it would be useful to know SCGF well before SCF, to enable prefetching of contexts. Prefetching contexts is necessary to hide the extra cycles to read from memory. Currently different sub-blocks may use different contexts based on the frequency region. If it is determined early on that a sub-block does not contain any significant\_coeffgroup\_flag, the contexts for that sub-block do not have to be pre-fetched.

SCGF

SCF

SCF

SCF

SCGF

SCF

SCF

SCF

SCGF

SCF

SCGF

SCF

Fig. 1 Signaling of significant\_coeffgroup\_flag (SCGF) and significant\_coeff\_flag (SCF) in HM-5.0

# Proposed Solution

This contribution proposes grouping the SCGF together and the SCF together and sending all the SCGF of a TU before the SCF as shown in Fig. 2. This will enable accurate prefetch of context for SCF and also minimize context switching between SCGF context and SCF contexts.

SCGF

SCF

SCF

SCF

SCGF

SCF

SCF

SCF

SCF

SCGF

SCGF

SCF

Fig. 1 Proposed signaling of significant\_coeffgroup\_flag (SCGF) and significant\_coeff\_flag (SCF)

# Working Draft Text

### Residual coding syntax

|  |  |
| --- | --- |
| residual\_coding\_cabac( x0, y0, log2TrafoWidth, log2TrafoHeight, scanIdx, cIdx ) { | Descriptor |
| **last\_significant\_coeff\_x\_prefix** | ae(v) |
| **last\_significant\_coeff\_y\_prefix** | ae(v) |
| if ( last\_significant\_coeff\_x\_prefix > 3 ) |  |
| **last\_significant\_coeff\_x\_suffix** | ae(v) |
| if ( last\_significant\_coeff\_y\_prefix > 3 ) |  |
| **last\_significant\_coeff\_y\_suffix** | ae(v) |
| numCoeff = 0 |  |
| do { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ numCoeff ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ numCoeff ][ 1 ] |  |
| numCoeff++ |  |
| } while( ( xC != LastSignificantCoeffX ) || ( yC != LastSignificantCoeffY ) ) |  |
| numLastSubset = (numCoeff − 1) >> 4 |  |
| for( i = numLastSubset; i >= 0; i− − ) { |  |
| offset = i << 4 |  |
| if( max( log2TrafoWidth, log2TrafoHeight ) > 3) { |  |
| xCG = ScanOrder[ log2TrafoWidth – 2 ][ log2TrafoHeight – 2 ][ scanIdx ][ i ][ 0 ] |  |
| yCG = ScanOrder[ log2TrafoWidth – 2 ][ log2TrafoHeight – 2 ][ scanIdx ][ i ][ 1 ] |  |
| rightCGFlag = (xCG = = (1<< (log2TrafoWidth – 2)) − 1) ? 0 :   significant\_coeff\_group\_flag[ xCG + 1 ][ yCG ] |  |
| bottomCGFlag = (yCG = = (1 << (log2TrafoHeight – 2)) − 1) ? 0 :   significant\_coeff\_group\_flag[ xCG ][ yCG + 1 ] |  |
| if( (i = = numLastSubset) | | (rightCGFlag + bottomCGFlag = = 2) | | (i = = 0) ) |  |
| significant\_coeff\_group\_flag[ xCG ][ yCG ] = 1 |  |
| else |  |
| **significant\_coeff\_group\_flag**[ xCG ][ yCG ] | ae(v) |
| } |  |
| } |  |
| for( i = numLastSubset; i >= 0; i− − ) { |  |
| offset = i << 4 |  |
| if( max( log2TrafoWidth, log2TrafoHeight ) > 3) { |  |
| xCG = ScanOrder[ log2TrafoWidth – 2 ][ log2TrafoHeight – 2 ][ scanIdx ][ I ][ 0 ] |  |
| yCG = ScanOrder[ log2TrafoWidth – 2 ][ log2TrafoHeight – 2 ][ scanIdx ][ i ][ 1 ] |  |
| if (significant\_coeff\_group\_flag[ xCG ][ yCG ] ) |  |
| { |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if ( (n + offset) < (numCoeff − 1) && significant\_coeff\_group\_flag[ xCG ][ yCG ] ) { |  |
| numNZInCG = (i = = numLastSubset) ? 1 : 0 |  |
| if ( (n > 0) | | (rightCGFlag + bottomCGFlag = = 2) | | (i = = 0) | | (numNZInCG > 0) ) { |  |
| **significant\_coeff\_flag**[ xC ][ yC ] | ae(v) |
| numNZInCG += significant\_coeff\_flag[ xC ][ yC ] |  |
| } |  |
| else |  |
| significant\_coeff\_flag[ xC ][ yC ] = 1 |  |
| } |  |
| } |  |
| } |  |
| } else { |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( (n + offset) < (numCoeff − 1) ) |  |
| **significant\_coeff\_flag[** xC **][** yC **]** | ae(v) |
| } |  |
| } |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] ) |  |
| **coeff\_abs\_level\_greater1\_flag[** n **]** | ae(v) |
| } |  |
| for( n = 15; n >= 0; n− − ) { |  |
| if( coeff\_abs\_level\_greater1\_flag[ n ] ) |  |
| **coeff\_abs\_level\_greater2\_flag[** n **]** | ae(v) |
| } |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] ) { |  |
| **coeff\_sign\_flag[** n **]** | ae(v) |
| } |  |
| for( n = 15; n >= 0; n− − ) { |  |
| if( coeff\_abs\_level\_greater2\_flag[ n ] ) |  |
| **coeff\_abs\_level\_minus3[** n **]** | ae(v) |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] ) { |  |
| transCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =   ( coeff\_abs\_level\_minus3[ n ] + 3 ) \* ( 1 − 2 \* coeff\_sign\_flag[ n ] ) |  |
| } else |  |
| transCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] = 0 |  |
| } |  |
| } |  |
| } |  |

# Patent rights declaration(s)

Texas Instruments Inc. may have IPR 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).