

Discussion on VVC Subpicture Support

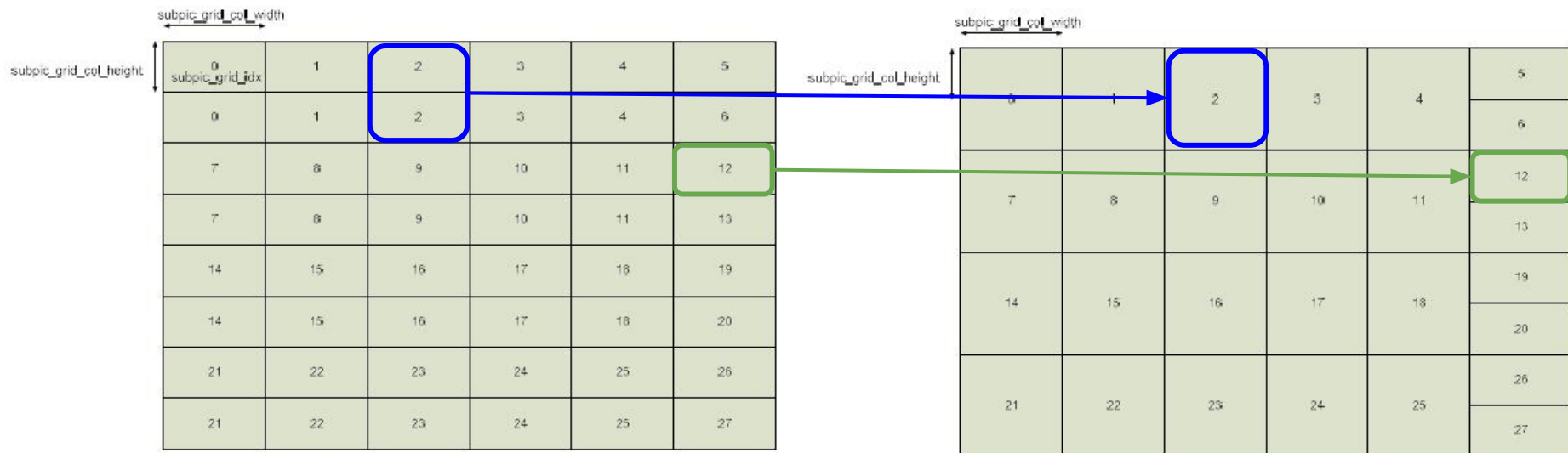
August 26, 2019

Background

- The signaling of subpicture boundaries has implications for decoder implementations (potential interop issues as well as storage and processing requirements)
- A subpicture grid approach was adopted at the last meeting based on JVET-O1143
- These slides are an analysis of the existing draft text in an attempt to:
 - Confirm understanding of the adoption and design philosophy
 - Provide feedback on the implementation implications
 - Motivate development of alternative approaches to reduce the implementation impact while still satisfying the general subpicture functionality

Overview of Current Subpicture Signaling

- Defines a brand new partitioning scheme using **subpic_grid_col_height** and **subpic_grid_col_width**
 - Each “block” in this grid has a transmitted **subpic_grid_idx**
 - Conceptually, these need to be “grouped together” to determine each subpicture and its boundaries



Concerns on Checking Conformance & Interop

- Semantics may constrain but not difficult from the syntax in the SPS to create illegal indexing, also hard to check conformance: **potential interop problems!**
 - Non-rectangular indexing
 - Grouping non-connected regions
 - Not aligned with slice boundaries

subpics_present_flag	u(1)
if(subpics_present_flag) {	
max_subpics_minus1	u(8)
subpic_grid_col_width_minus1	u(v)
subpic_grid_row_height_minus1	u(v)
for(i = 0; i < NumSubPicGridRows; i++)	
for(j = 0; j < NumSubPicGridCols; j++)	
subpic_grid_idx[i][j]	u(v)
for(i = 0; i <= NumSubPics; i++) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	

Overview of Current Subpicture Signaling

- Boundaries (top, left, width and height) need to be derived from this grid

```
NumSubPics = 0
for( i = 0; i < NumSubPicGridRows; i++ ) {
  for( j = 0; j < NumSubPicGridCols; j++ ) {
    if( i == 0 )
      SubPicTop[ subpic_grid_idx[ i ][ j ] ] = 0
    else if( subpic_grid_idx[ i ][ j ] != subpic_grid_idx[ i - 1 ][ j ] ) {
      SubPicTop[ subpic_grid_idx[ i ][ j ] ] = i
      SubPicHeight[ subpic_grid_idx[ i - 1 ][ j ] ] = i - SubPicTop[ subpic_grid_idx[ i - 1 ][ j ] ]
    }
    if( j == 0 )
      SubPicLeft[ subpic_grid_idx[ i ][ j ] ] = 0
    else if( subpic_grid_idx[ i ][ j ] != subpic_grid_idx[ i ][ j - 1 ] ) {
      SubPicLeft[ subpic_grid_idx[ i ][ j ] ] = j
      SubPicWidth[ subpic_grid_idx[ i ][ j ] ] = j - SubPicLeft[ subpic_grid_idx[ i ][ j - 1 ] ]
    }
    if( i == NumSubPicGridRows - 1 )
      SubPicHeight[ subpic_grid_idx[ i ][ j ] ] = i - SubPicTop[ subpic_grid_idx[ i - 1 ][ j ] ] + 1
    if( j == NumSubPicGridCols - 1 )
      SubPicWidth[ subpic_grid_idx[ i ][ j ] ] = j - SubPicLeft[ subpic_grid_idx[ i ][ j - 1 ] ] + 1
    if( subpic_grid_idx[ i ][ j ] > NumSubPics )
      NumSubPics = subpic_grid_idx[ i ][ j ]
  }
}
```

(7-7)

Side topic / minor issue:
Text is from JVET-O2001-vE
Believe “j” should be “j-1” in these
equations but have not confirmed yet

Storage costs

subpic_grid_idx[i][j] specifies the subpicture index of the grid position (i, j). The length of the syntax element is $\text{Ceil}(\text{Log2}(\text{max_subpics_minus1} + 1))$ bits.

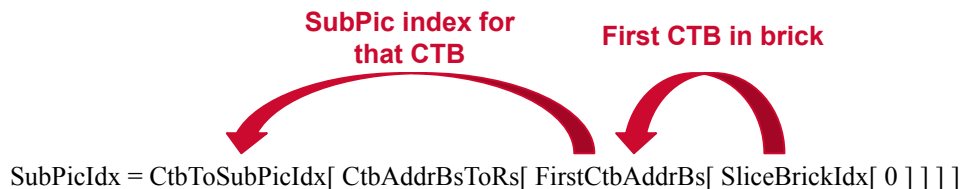
max_subpics_minus1 plus 1 specifies the maximum number of subpictures that may be present in the CVS.
max_subpics_minus1 shall be in the range of 0 to 254.

Note subpicture grid is allowed to be as small as 4 sample resolution
Unclear why grid resolution is not based on CTU size?

Video resolution	Worst case memory requirement <u>per SPS</u>
4K (4096x2160):	8 bits for grid_idx 552,960 elements 553 kB
8K (8192x4320)	8 bits for grid_idx 2,211,840 elements 2.21 MB

Overview of Current Subpicture Signaling

- Decoder needs to determine subpicture boundaries because it affects:
 - Motion Compensation
 - Clip TMVP positions
 - Deblocking can be disabled on these boundaries
- Every slice is mapped to a subpicture ID using multiple mapping tables:
 - Find first brick in slice
 - Find first CTB in that brick
 - Find SubPic index for that CTB



Overview of Current Subpicture Signaling

- An example of the processing required for one of these mappings:
- “Find SubPic index for that CTB”
 - Find top-left corner of current CTB
 - Find subpicture which contains this CTB by scanning through each subpicture and comparing boundaries until you find a match

```
for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++) {  
    posX = ctbAddrRs % PicWidthInCtbsY * CtbSizeY  
    posY = ctbAddrRs / PicWidthInCtbsY * CtbSizeY  
    CtbToSubPicIdx[ ctbAddrRs ] = -1  
    for( i = 0; CtbToSubPicIdx[ ctbAddrRs ] < 0 && i < NumSubPics; i++) {  
        if( (posX >= SubPicLeft[ i ] * (subpic_grid_col_width_minus1 + 1) * 4) &&  
            (posX < ( SubPicLeft[ i ] + SubPicWidth[ i ] ) *  
                (subpic_grid_col_width_minus1 + 1) * 4) &&  
            (posY >= SubPicTop[ i ] *  
                (subpic_grid_row_height_minus1 + 1) * 4) &&  
            (posY < ( SubPicTop[ i ] + SubPicHeight[ i ] ) *  
                (subpic_grid_row_height_minus1 + 1) * 4) )  
            CtbToSubPicIdx[ ctbAddrRs ] = i  
    }  
}
```

(6-11)

Discussion

- Was confirmed on reflector that design intent is for a subpicture to be composed of one or more complete slices
- Question: Why is a complex grid based approach defined with boundaries that can be arbitrarily aligned at a 4-pixel resolution?
 - It seems better to directly match subpicture definitions to existing structures rather than defining another new separate partitioning scheme.
 - Bricks, tiles and slices are already defined in terms of each other so it seems a natural and more intuitive approach is for subpictures to be defined by grouping slices.
- Approaches to consider:
 - Define a subpicture index for each slice. This could be a FLC at the top of the slice header for easy bit stream manipulation.
 - Associate slices to subpictures in one of the loops across the slices of a picture in the PPS