

JCTVC-J284

Tiles and Wavefront Parallel Processing Restrictions For The Main Profile

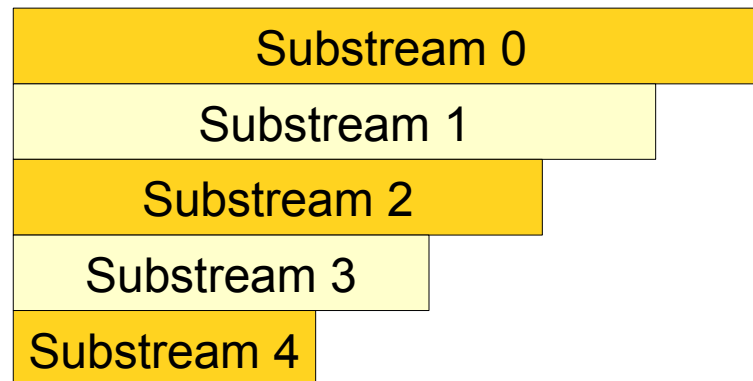
Stewart Worrall, Adrian Wise

Aspex Semiconductor

July 2012

- Both WPP and Tiles impose an additional burden on decoders
 - Parallel processing options should place as small a burden as possible on decoders
- Parallel decoding with a very wide scope of parallel configurations will be impractical
- Proposal:
 - Restrictions on WPP usage and tile formats to reduce decoder implementation burden, and make parallel decoding simpler

- WPP is already restricted to one substream per CTB row
 - This considerably simplifies decoder implementation
- Additional restrictions
 - Include entry point indicators for every substream if parallel decoding is important
 - Restrict WPP usage to levels 3.1 and above



Issues For Restricting Tile Size

- If we restrict tile sizes in the Main Profile, the following issues should be considered
 - MTU size matching
 - Parallel encoding/decoding flexibility
 - Compression efficiency
 - Decoder conformance testing
 - Level based settings
 - Line buffer issues

MTU Size Matching

- Some sequences (e.g. Class E) can have tiles that vary significantly in size
 - Some tiles are much less than one MTU size
 - Multiple tiles in one slice is the best solution
 - Some tiles are larger than one MTU size
 - Tiles need splitting into multiple slices
- Smaller tiles could allow all tiles to fit inside on MTU, making MTU slice size matching simpler
- Best tile size will vary with QP, making some flexibility desirable



- Parallel encoding/decoding flexibility
 - Tiles can be grouped and sent to different cores for processing
 - Need enough tiles to exploit different numbers of cores
 - Need to provide enough tiles to allow encoder and decoder implementers to be able to use different numbers of cores at a particular resolution



- Compression efficiency
 - Use of too many tiles reduces compression efficiency
 - Reasonable trade-off should be selected between compression efficiency and parallel processing flexibility
 - More tiles may allow more cycles per CTB => encoder can use better RDO algorithms
- Decoder conformance testing
 - Fewer possible configurations are preferred
 - Reduces conformance testing burden significantly
 - May reduce number of legal “evil” bitstreams
 - Fixed column widths would make implementation of raster scan order decoding of tiles more straightforward

- Level based settings
 - Advantage and disadvantage: Can be worked around
 - Currently, a 1080p bitstream can be expected to be coded at a particular level
 - Encoders may use a higher level to get more tiles
 - Complicates implementation
 - Customers may want a 1080p decoder, but what level does this mean if tile numbers are linked and limited by levels?
 - Will there be 1080p streams “in the wild” with many different level numbers?
- Line buffer issues
 - Implementation dependent
 - Relevant for encoders
 - Decoders will have to implement tiles and non-tiles versions
 - May not always be possible to achieve potential line buffer reductions

Proposed Limitations: Column Width



- Tiles should only be used for levels above 3.1
- `uniform_spacing_flag = 1`
 - Note: this would fix tile columns and widths
- Column width constraints
 - `ColumnWidthInLumaSamples[i]` shall be greater than or equal to 384 for any `i` in the range of 0 to `num_tile_columns_minus1`, inclusive
 - `num_tile_columns_minus1 = Floor(PicWidthInCtbs / (6 * 26-Log2CtbSize)) - 1`
 - This restricts the number of tile columns irrespective of CTB size
 - $64 * 6 = 384$

Proposed Limitations: Row Height



- BaseTileRows shall be derived as follows:
 - $\text{TileConstraintA} = \text{Ceil}(\text{PicHeightInCtbs} / 2^{6-\text{Log2CtbSize}} / (\text{num_tile_columns_minus1} + 1))$
 - $\text{TileConstraintB} = \text{Floor}(\text{PicHeightInCtbs} / (2^{6-\text{Log2CtbSize}} * 3))$
 - $\text{BaseTileRows} = \text{Min}(\text{TileConstraintA}, \text{TileConstraintB})$
- The following constraints shall be obeyed:
 - $\text{BaseTileRows} - 2 < \text{num_tile_rows_minus1} < \text{BaseTileRows} + 2$
- This has the effect of allowing three different tile configurations for any particular picture resolution
- Column widths are always fixed for a particular resolution
 - Simplifies raster scan decoding of a picture encoded with tiles

Full text available in attached document.

Main tile constraints implemented by following text:

When `tiles_or_entropy_coding_sync_idc` is equal to 1, the following constraints shall be obeyed:

- Picture Parameter sets shall have `uniform_spacing_flag` equal to 1.
- Picture Parameter sets shall have `num_tile_columns_minus1` equal to $\text{PicWidthInCtbs} / (6 * 2^{6-\text{Log2CtbSize}}) - 1$.
- Picture Parameter sets shall have `num_tile_rows_minus1` in the range of `BaseTileRows - 1` to `BaseTileRows + 1`, where `BaseTileRows` shall be derived as follows:

$$\text{TileConstraintA} = \text{Ceil}(\text{PicHeightInCtbs} / 2^{6-\text{Log2CtbSize}} / (\text{num_tile_columns_minus1} + 1))$$

$$\text{TileConstraintB} = \text{Floor}(\text{PicHeightInCtbs} / (2^{6-\text{Log2CtbSize}} * 3))$$

$$\text{BaseTileRows} = \text{Min}(\text{TileConstraintA}, \text{TileConstraintB})$$

Example Tile Specifications

- The proposed profile settings should provide the following example tile rows and columns
 - Spreadsheet included with contribution allows calculation of tile numbers at any input resolution

Class	LumaWidth	LumaHeight	num_tile_columns	BaseTileRows	Min Num Tiles	Med Num Tiles	Max Num Tiles
4K	3840	2160	10	4	30	40	50
A	2560	1600	6	4	24	30	36
B	1920	1080	5	4	15	20	25
C	832	480	2	2	2	4	6
E	1280	720	3	4	9	12	15

- WPP and tiles should both be restricted to allow easier decoder implementation
 - Parallel processing tools should only be allowed for levels above 3.1
 - WPP already has appropriate constraints (one substream per CTB row)
 - Tiles need restricting
- Tile height and width should be tightly constrained to allow simpler decoder implementation
 - Allow some flexibility in sizes to allow implementers to find good trade-offs between efficient load balancing and compression efficiency
- Proposed restrictions are in effect set according to picture resolution, rather than levels, which are more loosely linked with resolution