

REDEFINING MOBILITY



## JCTVC-K0258 I\_PCM Signalling

M. Coban, R. Joshi, M. Karczewicz

# I\_PCM Signalling

- Current WD 8 scheme (Burst I\_PCM):

- Signals starts of a PCM block (**pcm\_flag**) followed by signalling of number of subsequent I\_PCM blocks (0-3) (**num\_subsequent\_pcm**) that follow the first I\_PCM unit at a layer of CTB.
- Main motivation: increase throughput of PCM data while reducing side information bits [JCTVC-H0051]

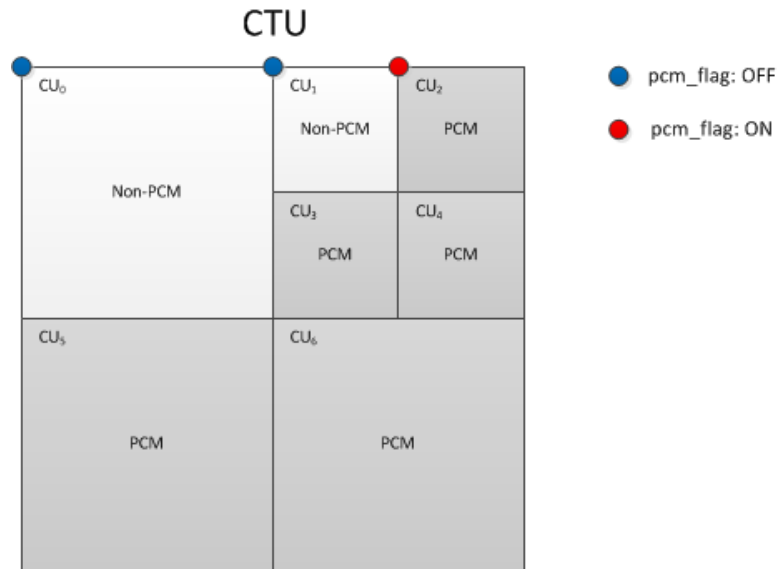
- Motivation:

- Bit constraint for CABAC is defined at CTU level.
- HW CABAC engines typically operate at the granularity of a CTU or higher
- I\_PCM mode is applied at CTU level in order to meet the bit constraint or cycle budget to encode the bits
- Once a switch to I\_PCM mode is made at a CU, I\_PCM mode would be used until the end of the CTU

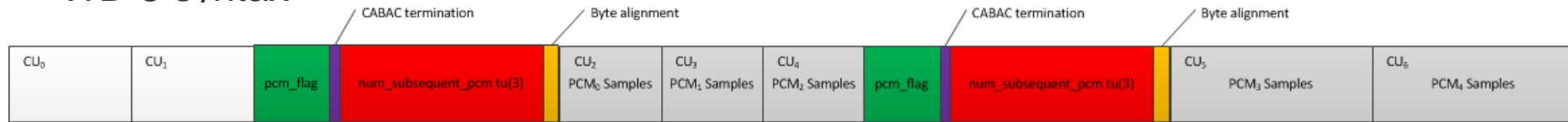
- Proposal:

- Use the existing pcm\_flag to signal the starting point of I\_PCM samples within a CTU that lasts until the end of the CTU.

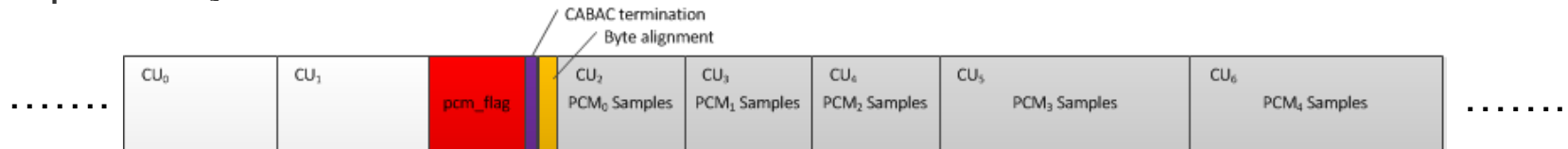
# Proposed syntax



## ■ WD 8 syntax



## ■ Proposed syntax



# Results: BD Rate (JCTVC-J1100)

	Y	U	V
AI-Main	0.0%	0.0%	0.0%
RA-Main	0.0%	0.0%	0.0%
LB-Main	0.0%	0.0%	0.0%

Anchor: HM 8.0 with I\_PCM enabled

CTC: I\_PCM is not chosen

# Conclusion

- Simpler, more CABAC friendly I\_PCM signalling using only existing **pcm\_flag** syntax
- Recommend adoption of this proposal.
- We thank MediaTek for cross-checking (JCTVC-K0300).

# WD changes

coding_tree_unit( xCtb, yCtb ) {	Descriptor
NumPCMBBlock = 0	
PCMFlag = 0	
xCtb = InverseRasterScan( CtbAddrRS, CtbSize, CtbSize, pic_width_in_luma_samples, 0 )	
.....	

coding_quadtree( x0, y0, log2CbSize, ctDepth ) {	Descriptor
if( x0 + ( 1 << log2CbSize ) <= pic_width_in_luma_samples && y0 + ( 1 << log2CbSize ) <= pic_height_in_luma_samples && log2CbSize > Log2MinCbSize && PCMFlag NumPCMBBlock == 0 )	
split_cu_flag[ x0 ][ y0 ]	ae(v)
.....	
} else {	
if( PCMFlag NumPCMBBlock == 0 )	
coding_unit( x0, y0, log2CbSize )	
else	
pcm_sample( x0, y0, log2CbSize )	
}	
}	

coding_unit( x0, y0, log2CbSize ) {	Descriptor
....	
if( PredMode[ x0 ][ y0 ] == MODE_INTRA ) {	
if( PartMode == PART_2Nx2N && pcm_enabled_flag && log2CbSize >= Log2MinIPCMCUSize && log2CbSize <= Log2MaxIPCMCUSize )	
pcm_flag	ae(v)
if( pcm_flag ) {	
PCMFlag = 1	
num_subsequent_pcm	tu(3)
NumPCMBBlock = num_subsequent_pcm + 1	
while( !byte_aligned( ) )	
pcm_alignment_zero_bit	f(1)
pcm_sample( x0, y0, log2CbSize )	
} else {	
.....	

# WD changes

## 9.1 Parsing process for Truncated Unary codes

This process is invoked when the descriptor of a syntax element in the syntax tables in subclause 7.3 is equal to tu(n).

Inputs to this process are bits from the RBSP.

Outputs of this process are syntax element values.

Syntax elements coded as tu(n) are truncated unary coded. The range of possible values for the syntax element is determined first. The range of this syntax element may be between 0 and n, with n being greater than or equal to 1 and the range is used in the derivation of the value of the syntax element value. codeNum which is equal to the value of the syntax element is given by a process specified as follows:

```
codeNum = 0
keepGoing = 1
for(i = 0; i < n && keepGoing; i++){
    keepGoing = read_bits( 1 ) (9-1)
    if( keepGoing )
        codeNum ++
}
```

■

# WD changes

**split\_cu\_flag**[ x0 ][ y0 ] specifies whether a coding unit is split into coding units with half horizontal and vertical size. The array indices x0, y0 specify the location ( x0, y0 ) of the top-left luma sample of the considered coding block relative to the top-left luma sample of the picture.

When **split\_cu\_flag**[ x0 ][ y0 ] is not present, the following applies:

- If log2CbSize is greater than Log2MinCbSizeY **and PCMFlag is equal to 0**, the value of **split\_cu\_flag**[ x0 ][ y0 ] is inferred to be equal to 1.
- Otherwise (log2CbSize is equal to Log2MinCbSizeY **or the current PCMFlag is not equal to 0**), the value of **split\_cu\_flag**[ x0 ][ y0 ] is inferred to be equal to 0.