

# Simplification on zero merge candidate derivation

Hui Yong Kim

# Summary

## □ Proposal

- ❖ Zero merge candidate derivation process regardless of slice\_type

## □ Benefits

- ❖ Simplified design
  - Unified zero merge candidate derivation process for P-slice and B-slice
- ❖ Simplified text
  - 1/3 of text in “8.5.2.1.4 Derivation process for zero motion vector merging candidates” is removed.
- ❖ Reduced MC complexity
  - MC complexity for B-slice is reduced when zero merge candidate is selected.

## □ Results

- ❖ No coding loss

# HM 7.0

## ❑ Zero merge candidate derivation in merge mode

1. Zero merge candidate addition **depends on slice\_type**
  - For P-slice, L0 uni-predictive zero merge candidate (increasing refIdx by 1)
  - For B-slice, Bi-predictive zero merge candidate (increasing refIdx by 1)
2. Zero merge candidate fill-up **depends on slice\_type**
  - For P-slice, L0 uni-predictive zero merge candidate (refIdx == 0)
  - For B-slice, Bi-predictive zero merge candidate (refIdx == 0)

Variable	1) Zero merge candidate addition process		2) Zero merge candidate fill-up process	
	P-slice	B-slice	P-slice	B-slice
predFlagL0	1	1	1	1
predFlagL1	0	1	0	1
mvL0	(0, 0)	(0, 0)	(0, 0)	(0, 0)
mvL1	(0, 0)	(0, 0)	(0, 0)	(0, 0)
refIdxL0	r	r	0	0
refIdxL1	-1	r	-1	0

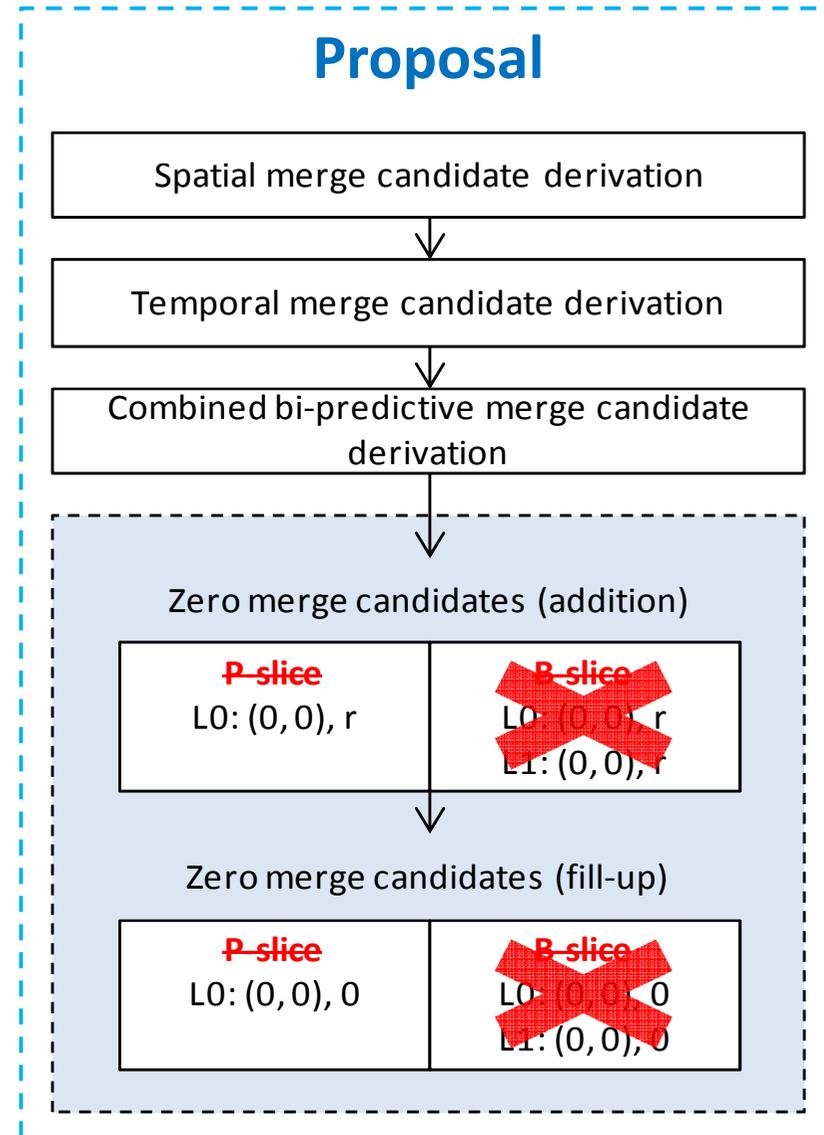
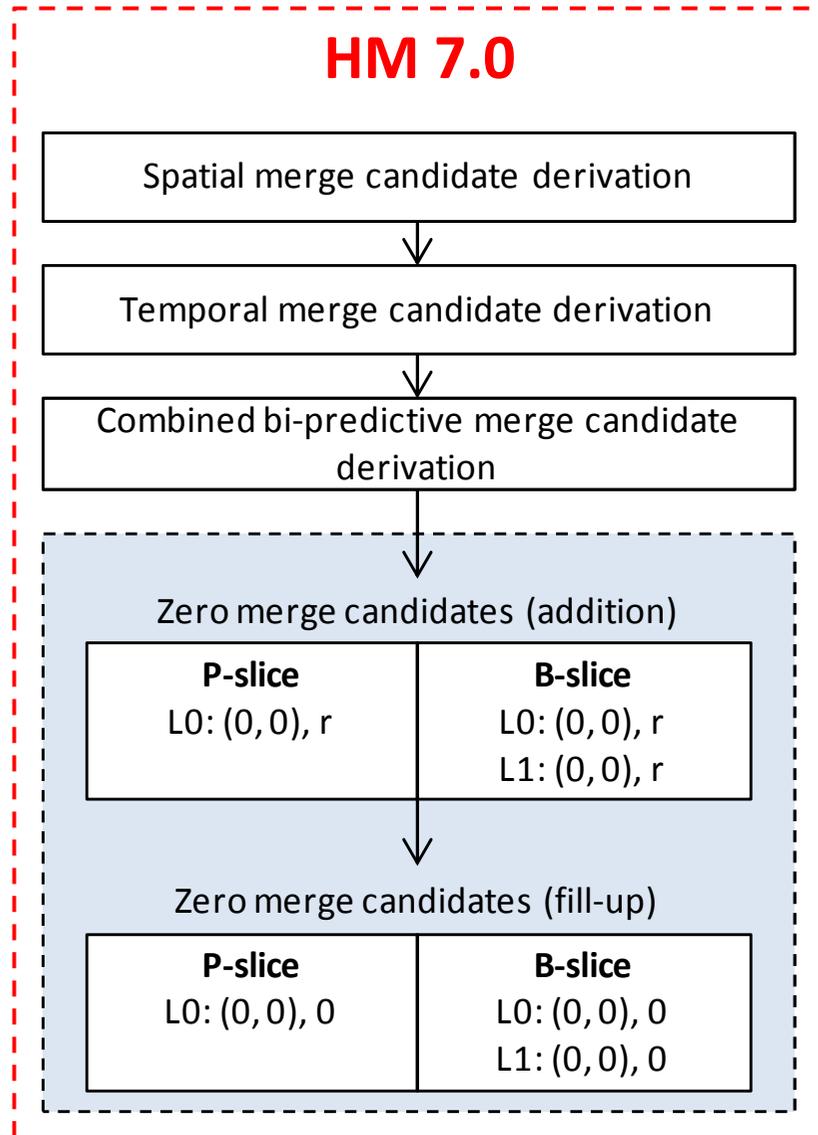
# Proposal

## ❑ Use L0 uni-predictive zero merge candidate for both P-slice and B-slice

1. Zero merge candidate addition **regardless of slice\_type**
  - L0 uni-predictive zero merge candidate (increasing refIdx by 1)
2. Zero merge candidate fill-up **regardless of slice\_type**
  - L0 uni-predictive zero merge candidate (refIdx == 0)

Variable	1) Zero merge candidate addition process	2) Zero merge candidate fill-up process
predFlagL0	1	1
predFlagL1	0	0
mvL0	(0, 0)	(0, 0)
mvL1	(0, 0)	(0, 0)
refIdxL0	r	0
refIdxL1	-1	-1

# Comparison



# Benefits

## ❑ Simplified design

- ❖ Unified zero merge candidate derivation process for P-slice and B-slice

## ❑ Simplified text

- ❖ No additional line is added.
- ❖ 1/3 of text in “8.5.2.1.4 Derivation process for zero motion vector merging candidates” is removed.

## ❑ Reduced MC complexity

- ❖ MC complexity for B-slice is reduced when zero merge candidate is selected.

### 8.5.2.1.4 Derivation process for zero motion vector merging candidates:

Inputs of this process are

- a merging candidate list `mergeCandList`,
- reference indices `refIdxL0N` and `refIdxL1N` of every candidate `N` being in `mergeCandList`,
- prediction list utilization flags `predFlagL0N` and `predFlagL1N` of every candidate `N` being in `mergeCandList`,
- motion vectors `mvL0N` and `mvL1N` of every candidate `N` being in `mergeCandList`,
- the number of elements `numMergeCand` within `mergeCandList`.

Outputs of this process are

- the merging candidate list `mergeCandList`,
- the number of elements `numMergeCand` within `mergeCandList`,
- reference indices `refIdxL0zeroCandn` and `refIdxL1zeroCandn` of every new candidate `zeroCandn`, being added in `mergeCandList` during the invocation of this process,
- prediction list utilization flags `predFlagL0zeroCandn` and `predFlagL1zeroCandn` of every new candidate `zeroCandn`, being added in `mergeCandList` during the invocation of this process,
- motion vectors `mvL0zeroCandn` and `mvL1zeroCandn` of every new candidate `zeroCandn`, being added in `mergeCandList` during the invocation of this process.

The variable `numRefIdx` is defined as follows:

- ~~If slice\_type is equal to P, `numRefIdx` is set to `num_ref_idx_l0_active_minus1 + 1`.~~
  - ~~Otherwise (slice\_type is equal to B), `numRefIdx` is set to `min(num_ref_idx_l0_active_minus1 + 1, num_ref_idx_l1_active_minus1 + 1)`.~~
- When `numMergeCand` is less than `MaxNumMergeCand`, the variable `numInputMergeCand` is set to `numMergeCand`, the variable `zeroIdx` is set to 0, and the following steps are repeated until `numMergeCand` is equal to `MaxNumMergeCand`.

1. For the derivation of the reference indices, the prediction list utilization flags and the motion vectors of the zero motion vector merging candidate, the following applies:

- ~~If slice\_type is equal to B, the candidate `zeroCandn`, with `n` equal to `(numMergeCand - numInputMergeCand)` is added at the end of `mergeCandList` (`mergeCandList[numMergeCand] = zeroCandn`) and the reference indices, the prediction list utilization flags and the motion vectors of `zeroCandn` are derived as follows and `numMergeCand` is incremented by 1.~~
- |  |                    |
|--|--------------------|
| <del><code>refIdxL0zeroCand<sub>n</sub> = (zeroIdx - numRefIdx) ? zeroIdx : 0</code></del> | <del>(8-99)</del>  |
| <del><code>refIdxL1zeroCand<sub>n</sub> = -1</code></del>                                  | <del>(8-100)</del> |
| <del><code>predFlagL0zeroCand<sub>n</sub> = 1</code></del>                                 | <del>(8-101)</del> |
| <del><code>predFlagL1zeroCand<sub>n</sub> = 0</code></del>                                 | <del>(8-102)</del> |
| <del><code>mvL0zeroCand<sub>n</sub>[ 0 ] = 0</code></del>                                  | <del>(8-103)</del> |
| <del><code>mvL0zeroCand<sub>n</sub>[ 1 ] = 0</code></del>                                  | <del>(8-104)</del> |
| <del><code>mvL1zeroCand<sub>n</sub>[ 0 ] = 0</code></del>                                  | <del>(8-105)</del> |
| <del><code>mvL1zeroCand<sub>n</sub>[ 1 ] = 0</code></del>                                  | <del>(8-106)</del> |
| <del><code>numMergeCand = numMergeCand + 1</code></del>                                    | <del>(8-107)</del> |

- ~~Otherwise (slice\_type is equal to P), the candidate `zeroCandn`, with `n` equal to `(numMergeCand - numInputMergeCand)` is added at the end of `mergeCandList` (`mergeCandList[numMergeCand] = zeroCandn`) and the reference indices, the prediction list utilization flags and the motion vectors of `zeroCandn` are derived as follows and `numMergeCand` is incremented by 1:~~

- |  |                    |
|--|--------------------|
| <del><code>refIdxL0zeroCand<sub>n</sub> = (zeroIdx - numRefIdx) ? zeroIdx : 0</code></del> | <del>(8-108)</del> |
| <del><code>refIdxL1zeroCand<sub>n</sub> = (zeroIdx - numRefIdx) ? zeroIdx : 0</code></del> | <del>(8-109)</del> |
| <del><code>predFlagL0zeroCand<sub>n</sub> = 1</code></del>                                 | <del>(8-110)</del> |
| <del><code>predFlagL1zeroCand<sub>n</sub> = 1</code></del>                                 | <del>(8-111)</del> |
| <del><code>mvL0zeroCand<sub>n</sub>[ 0 ] = 0</code></del>                                  | <del>(8-112)</del> |
| <del><code>mvL0zeroCand<sub>n</sub>[ 1 ] = 0</code></del>                                  | <del>(8-113)</del> |
| <del><code>mvL1zeroCand<sub>n</sub>[ 0 ] = 0</code></del>                                  | <del>(8-114)</del> |
| <del><code>mvL1zeroCand<sub>n</sub>[ 1 ] = 0</code></del>                                  | <del>(8-115)</del> |
| <del><code>numMergeCand = numMergeCand + 1</code></del>                                    | <del>(8-116)</del> |

2. The variable `zeroIdx` is incremented by 1.

# Experimental results

## □ Results

❖ No coding loss (0.0%) on average in all common test conditions

Confirmed by  
Panasonic  
(JCTVC-J0443)

	Random Access Main			Random Access HE10		
	Y	U	V	Y	U	V
Class A	0.0%	-0.3%	-0.2%	0.0%	0.1%	0.1%
Class B	0.0%	0.0%	0.1%	0.1%	0.1%	0.0%
Class C	0.0%	0.0%	0.0%	0.0%	0.0%	-0.1%
Class D	0.0%	0.1%	-0.1%	0.0%	0.0%	-0.1%
Class E						
<b>Overall</b>	0.0%	0.0%	-0.1%	0.0%	0.1%	0.0%
	0.0%	0.0%	-0.1%	0.0%	0.1%	0.0%
Class F	0.0%	0.1%	0.0%	0.0%	0.1%	0.1%
Enc Time[%]		100%			98%	
Dec Time[%]		100%			100%	

	Low delay B Main			Low delay B HE10		
	Y	U	V	Y	U	V
Class A						
Class B	0.0%	0.0%	0.1%	0.0%	0.0%	0.4%
Class C	0.0%	0.3%	0.0%	0.0%	-0.1%	0.0%
Class D	0.0%	0.2%	0.0%	-0.1%	0.0%	0.2%
Class E	-0.1%	-0.2%	-0.5%	0.2%	0.1%	0.3%
<b>Overall</b>	0.0%	0.1%	-0.1%	0.0%	0.0%	0.2%
	0.0%	0.0%	0.0%	0.0%	0.0%	0.3%
Class F	-0.1%	-0.1%	0.0%	0.1%	0.0%	1.1%
Enc Time[%]		99%			97%	
Dec Time[%]		100%			100%	

# Conclusions

- ❑ **Proposal**
  - ❖ Zero merge candidate derivation process regardless of slice\_type
  
- ❑ **Benefits**
  - ❖ Simplified design, simplified text, reduced MC complexity
  
- ❑ **Source code modification**
  - ❖ Replacement of only one line and removal of five lines
  
- ❑ **Results**
  - ❖ No coding loss
  
- ❑ **Identical to JCTVC-J0180 (USTC & Huawei)**
  
- ❑ **We suggest the proposal to be included in DIS.**



***Thank You Very Much !***

[www.etri.re.kr](http://www.etri.re.kr)