|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  10th Meeting: Stockholm, Sweden, July 11-20, 2012 | Document: JCTVC-J0266 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **AHG6: Modification to loop filtering across slice boundaries** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Semih Esenlik,  Matthias Narroschke,  Thomas Wedi | Tel: Email: | +49 6103 766 - 1306 semih.esenlik@eu.panasonic.com |
| *Source:* | Panasonic Corp. | | |

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

# Abstract

In the current HEVC design *slice\_loop\_filter\_across\_slices\_enabled\_flag* in the slice header is used to control the loop filtering operations at the slice boundaries. The syntax element enables use cases such as “Gradual Decoder Refresh” (GDR) for which it is desirable to control the loop filters across the boundaries of each slice independently. However the interpretation of the syntax element is different for the deblocking Filter (DF) and SAO&ALF. The syntax element is used to control the top and left slice boundaries in the case of DF whereas all 4 boundaries (top, left, bottom and right) are controlled by the same flag in the case of ALF and SAO.

The proposal advocates aligning of the loop filtering operation at the slice boundaries such that all 3 loop filters are controlled at the top and left slice boundaries. According to the proposal the deblocking filtering control at the slice boundaries is not changed. ALF and SAO filtering control is modified such that SAO and ALF padding operations are applied at the same slice boundaries where deblocking filter is switched on or off.

# Introduction

The tables 1 and 2 below describe the syntax and semantics for *slice\_loop\_filter\_across\_slices\_enabled\_flag* in the slice header according to [2].

|  |  |
| --- | --- |
| slice\_header( ) { | Descriptor |
| ... |  |
| if( seq\_loop\_filter\_across\_slices\_enabled\_flag &&  ( slice\_adaptive\_loop\_filter\_flag | | slice\_sample\_adaptive\_offset\_flag | |  !disable\_deblocking\_filter\_flag ) ) |  |
| **slice\_loop\_filter\_across\_slices\_enabled\_flag** | u(1) |

**Table 1: Syntax for loop filter control across boundaries in slice header**

|  |
| --- |
| **slice\_loop\_filter\_across\_slices\_enabled\_flag** equal to 1 specifies that in-loop filtering operations are performed across slice boundaries; otherwise, the in-loop operations are slice-independent and not applied across slice boundaries. The in-loop filtering operations include the deblocking filter, sample adaptive offset filter, and adaptive loop filter. When slice\_loop\_filter\_across\_slices\_enabled\_flag is not present, it is inferred to be equal to seq\_loop\_filter\_across\_slices\_enabled\_flag. |

**Table 2: Semantics for the syntax element slice\_loop\_filter\_across\_slices\_enabled\_flag**

The syntax element was introduced in the slice header according to the contribution [1] during the 8th JCT-VC meeting in San Jose. The flag provides a mechanism to disable loop filters across the slice boundaries independently for each slice. One intended use case for *slice\_loop\_filter\_across\_slices\_enabled\_flag* in the slice header is “Gradual Decoder Refresh”. In this case, it is desirable to disable loop filters at the slice boundaries which separate the “refreshed” region from the “non-refreshed” region; hence a refreshed region in the frame does not depend on non-refreshed data. It should also be noted that a similar functionality exists also in the H264/AVC standard.

## Control of DF across slice boundaries

In the current HEVC CD, the *slice\_loop\_filter\_across\_slices\_enabled\_flag* is used to control the deblocking filtering at the top and left slice boundaries. If the *slice\_loop\_filter\_across\_slices\_enabled\_flag* is equal to 0, then top and left slice boundaries are not deblocked. This is illustrated in the figure below.

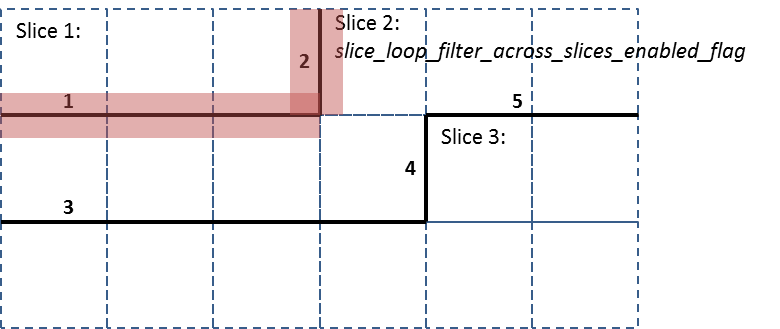


Figure : Control of deblocking filtering at slice boundaries. If *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 is equal to 0, deblocking is disabled at the shaded region.

Figure 1 presents an example for deblocking filter control at slice boundaries, in which *slice\_loop\_filter\_across\_slices\_enabled\_flag* is assumed to be equal to 0 in slice 2 and 1 in the other slices. According to the current HEVC design, which is described in [2], the deblocking filter is disabled at the shaded region (slice boundaries 1 and 2 in the figure).

## Control of SAO and ALF across slice boundaries

According to the HEVC text specification draft 7[2] and the HM7.1 software [3] the *slice\_loop\_filter\_across\_slices\_enabled\_flag* is used to control the ALF and SAO operations at the top, left, bottom and right slice boundaries.

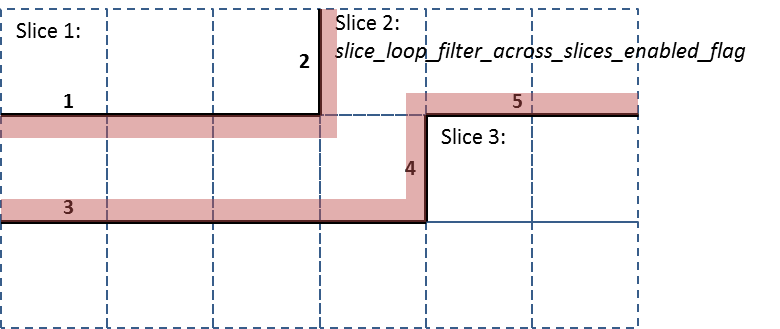


Figure : Control of ALF and SAO at slice boundaries. If *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 is equal to 0, SAO is disabled and ALF use padding at the shaded region.

The figure above presents an example for SAO and ALF control at slice boundaries. According to [2];

1. If *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 is equal to 0, then slice 2 is processed by SAO independently from slices 1 and 3. In other words, if a sample from slices 1 and 3 is required for the SAO processing of a sample in slice 2, then the edgeIdx is set to 0 for the sample in slice 2.
2. If *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 is equal to 0, then slice 2 is processed by ALF independently from slices 1 and 3. In other words, the boundaries of slice 2 (edges 1 to 5 in the above figure) are padded outwards. As a result samples from slices 1 and 3 are not required during the processing of slice 2 with ALF.

# Problem

# The below figure illustrates the inconsistency in the loop filtering control across the slice boundaries.

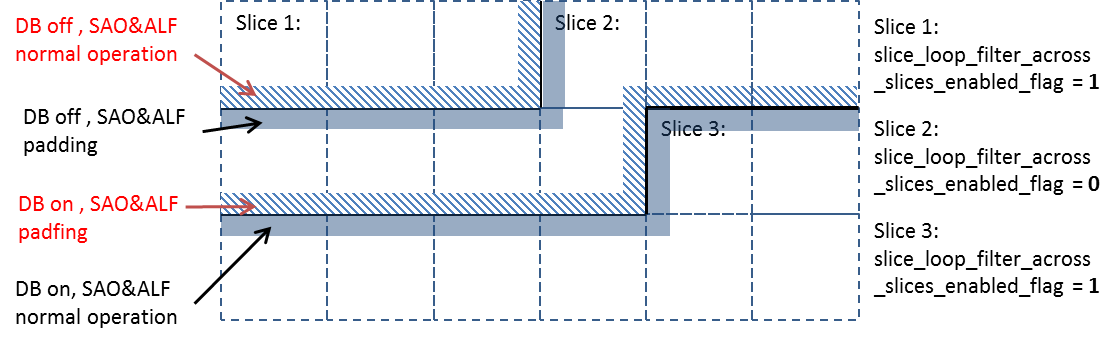


Figure : Illustration of the inconsistency of loop filter application across the slice boundaries. *slice\_loop\_filter\_across\_slices\_enabled\_flag* is assumed to be equal to 1 for slices 1 and 3, and 0 for slice 2.

In the example above *slice\_loop\_filter\_across\_slices\_enabled\_flag* is assumed to be equal to 1 for slices 1 and 3, and 0 for slice 2. According to the example:

1. At the top/left boundary of slice 2:

Deblocking filtering is disabled avoiding information dependency between slices 1 and 2. However SAO and ALF follow the normal operation at the upper side of the boundary since *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 1 is equal to 1. As a result the ALF and SAO create information dependency between slices 1 and 2.

1. At the bottom/right boundary of slice 2:

Deblocking filtering is enabled. However SAO is disabled and ALF uses padding at the upper side of the boundary since *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 is equal to 0.

The example in Figure 3 shows that setting the *slice\_loop\_filter\_across\_slices\_enabled\_flag* equal to 0 in slice 2 does not prevent the dependency between slices. Therefore during the gradual decoder refresh operation it is not possible to disable the loop filters across the slice boundaries in order to prevent information dependency between the slices.

The problem is emphasized especially in the case of top-to-down gradual decoder refresh operation (the most common GDR implementation) where slices in a frame are refreshed starting from the top-left corner of a frame. In such a case if slice 1 (slice no according to the coding order) is a refreshed slice and is proceeded by slice 2 which is a non-refreshed slice, then it is desirable to set the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice 2 to 0 and disable all three loop filters between the two slices. However according to the current HEVC design, during the processing of slice 1 SAO and ALF operations still use samples from slice 2 (meaning that the refreshed slice is contaminated with the non-refreshed data). Therefore the requirement of “independent refreshed and non-refreshed regions” cannot be satisfied.

Please note that the problem that is described above exists only for the SAO and ALF operations in the current HEVC CD. The deblocking operation of HEVC follows the boundary filtering control mechanism of H264/AVC in spirit. In other words only SAO and ALF cause conflict with respect to the requirements of the GDR operation.

# Proposal

The proposal advocates aligning all three loop filters by modifying the SAO and ALF slice boundary control operation. It is proposed that all three loop filters are controlled jointly at the top/left slice boundaries by the *slice\_loop\_filter\_across\_slices\_enabled\_flag* in the slice header and not at the bottom/right slice boundaries. In other words If *slice\_loop\_filter\_across\_slices\_enabled\_flag* is equal to 0 in a slice, then the following are applied according to the proposal:

1. Deblocking is disabled at both sides of the top/left slice boundary. (No change with respect to current HEVC [2])
2. SAO is controlled at both sides of the top/left slice boundary.
3. Padding is used for ALF at both sides of the top/left slice boundary.

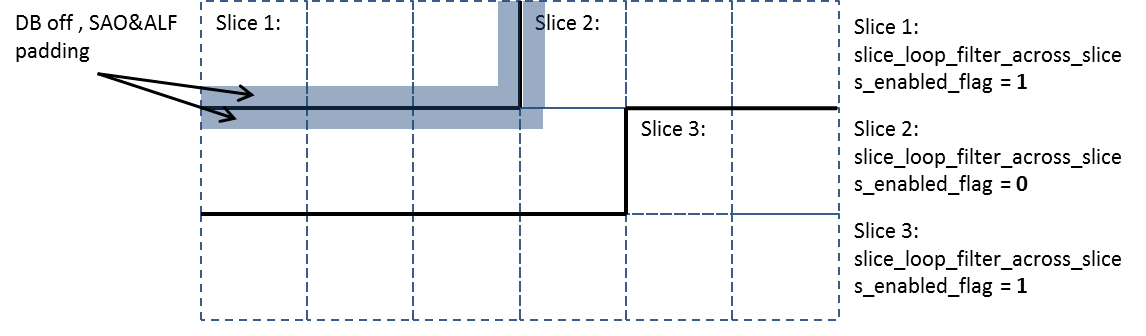


Figure : Proposed loop filtering control at the slice boundaries. All three loop filters are controlled at the top/left slice boundaries.

According to the proposal the SAO and ALF are aligned with the operation of deblocking filter, which is controlled only at the top and left boundaries of a slice. No change is proposed for the deblocking operation. Moreover no change is proposed for the specific padding operations of the ALF and SAO. It should also be noted that the proposed modification follows the H264/AVC in spirit, where deblocking filter is controlled only at the top and left slice boundaries.

The following two figures describe the details of the proposal for ALF and SAO. During the SAO and ALF operations in a slice, all Slice Granularity Units (SGUs) that are adjacent to the current slice are marked either “available for reference” or “unavailable for reference”. After the marking process is complete, ALF and SAO are applied to the current slice in a way that unavailable SGUs are not referenced in ALF and SAO processes. The Figure 5 and Figure 6 describe how the marking process is performed according to the proposal. (In the examples depicted in Figure 5and Figure 6 the smallest SGU corresponds to an LCU.)

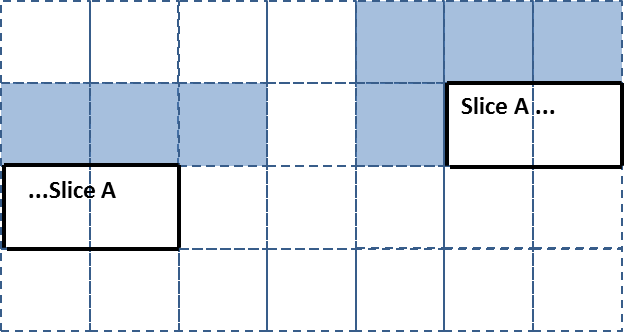
. 

Figure : Proposed modifications to the loop filtering control across slices. If the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice A is equal to 0, then the blue LCUs on the figure are marked as unavailable for reference for loop filtering.

Figure 5 above shows how the top and left neighboring LCUs are marked (if SGU size is smaller than the LCU size, the marking process applies to each SGU) according to the proposal. The top and left neighboring LCUs (blue LCUs in the figure) of slice A are set available for reference by slice A if the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice A is equal to 1. If the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice A is equal to 0, then all of the blue LCUs are marked as unavailable for reference by slice A.

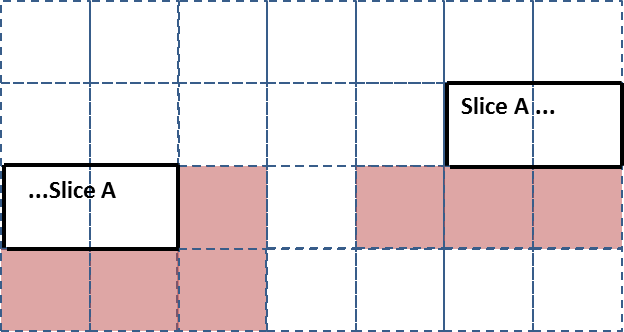


Figure : Proposed modifications to the loop filtering control across slices. The pink LCUs on the figure are marked as unavailable depending on the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of a right or bottom neighbor slice.

Figure 6 above describes the proposed marking process for the bottom and right neighbor LCUs of slice A (pink LCUs in the figure). If a bottom or right neighbor LCU (which is one of the pink LCUs in the figure) belongs to a slice in which the *slice\_loop\_filter\_across\_slices\_enabled\_flag* is equal to 1, then the LCU is marked as available for reference by slice A. Otherwise the LCU is marked as unavailable for reference by slice A.

The SAO and ALF operations are performed after the marking process is complete. During the SAO and ALF operations, the samples belonging to the SGUs that are marked as available for reference are used by ALF and SAO operations. The samples that belong to SGUs that are marked as unavailable for reference are not referred by SAO or ALF, instead usual boundary padding operations that are described in[2] for each tool are applied.

The proposal can be summarized by comparing it with the current HEVC design according [2] as follows:

**According to the current HEVC design:**

* All blue and pink LCUs in Figure 5and Figure 6 are marked available for reference by slice A, if the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice A is equal to 1. They are marked as unavailable for reference otherwise.

**According to the proposal:**

* All blue LCUs in Figure 5 are marked as available for reference by slice A if the *slice\_loop\_filter\_across\_slices\_enabled\_flag* of slice A is equal to 1. They are marked as unavailable for reference otherwise.
* If a pink LCU in Figure 6 belongs to a slice for which the *slice\_loop\_filter\_across\_slices\_enabled\_flag* is equal to 1, the LCU is marked as available for reference by slice A. It is marked as unavailable for reference otherwise

The description of the proposed changes is presented in the document “CD\_text\_modifications.doc”.

The proposed changes are implemented on the HM7.1 software [3]. The changes in the source code are encapsulated within the MODIFIED\_CROSS\_SLICE macro. In the software 16 lines of code has been changed (12 lines of which are removals.).

# Simulation Results

For the simulations the tagged version of HM7.1 has been used [3]. For the simulations the following encoder settings are used in order to enable multiple slices per picture:

SliceMode : 2

SliceArgument : 1500

LFCrossSliceBoundaryFlag : 0

Therefore according to the encoder settings 1500Byte slices are used and the loop filtering operations are not allowed to cross the slice boundaries. The following table shows the coding gain and runtime measurements for the proposal.



Table 3: Simulation results for the proposal

According to the simulation conditions loop filtering across slices are switched off for all of the slices. Please note that in this case the proposal and the current HEVC design behave the same way. In other words loop filtering is switched off across all of the slice boundaries in the proposal and HEVC. Therefore the results provided above verify that the proposal does not have any effect on the bitrate.

# Conclusions

The contribution advocates a modification in the control of three loop filters across the slice boundaries. According to the proposal the ALF and SAO slice boundary control operation is modified to align with the deblocking filtering control operation at the slice boundaries in order to facilitate gradual decoder refresh operation. It is proposed that the *slice\_loop\_filter\_across\_slices\_enabled\_flag* in the slice header is used to control all three loop filters jointly and only at the top/left slice boundaries. According to the proposal the control of deblocking filter at the slice boundaries is not changed. ALF and SAO filtering control is modified such that SAO and ALF padding operations are applied at the same slice boundaries where deblocking filter is switched on or off by the *slice\_loop\_filter\_across\_slices\_enabled\_flag*.

# References

1. R. Srinivasan, C. Ghone, “AHG15: Slice-Level Control of In-Loop Filter”, JCT-VC Document, JCTVC-H0391, 8th Meeting: San Jose, USA, January, 2012.

1. [B. Bross](mailto:benjamin.bross@hhi.fraunhofer.de), [W.-J. Han](mailto:wjhan.han@samsung.com), [J.-R. Ohm](mailto:ohm@ient.rwth-aachen.de), [G. J. Sullivan](mailto:garysull@microsoft.com), [T. Wiegand](mailto:thomas.wiegand@hhi.fraunhofer.de) “High Efficiency Video Coding (HEVC) text specification draft 7,” JCT-VC Document, JCTVC-I1003\_d6, 9th Meeting: Geneva, Switzerland, 27 April – 07 May, 2012.
2. HM7.1 Software available under the website: http://hevc.kw.bbc.co.uk/trac/browser/tags/HM-7.1

# Patent rights declaration(s)

**Panasonic Corporation may have current or pending patent rights 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).**