

AHG9: On LCU bit size limit (JCTVC-K0176)

Keiichi Chono
NEC Corporation

Background

Last JCTVC meeting

- Limit to 2x expansion in bits at LCU level was adopted.
 - JCTVC-J0057 proposed the limit of 1.25x expansion.
 - JCTVC-J0082 proposed the limit of 1.33x expansion.
- Further study was encouraged to investigate whether this should be lowered.

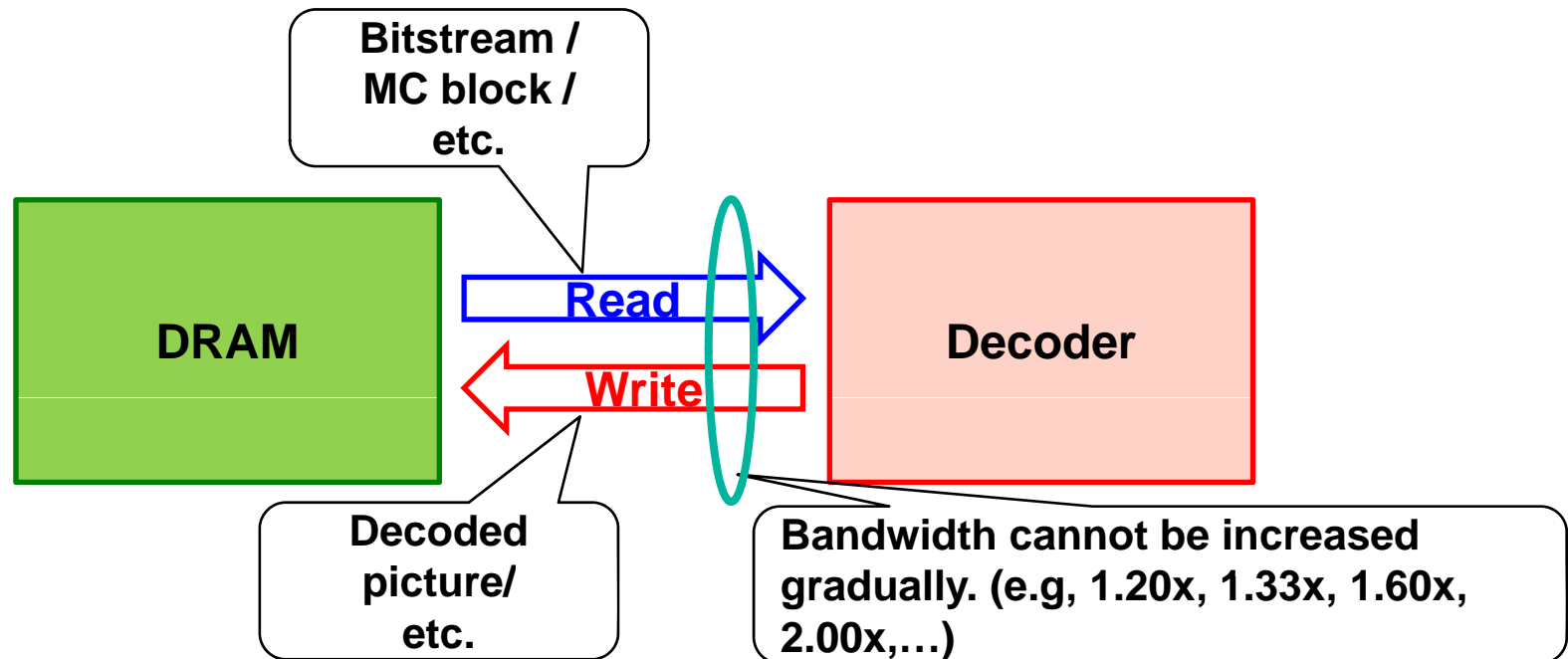
This contribution investigates the LCU bit size limit from the viewpoint of peak data transfer rates between DRAM and decoder.



Data transfer between DRAM and decoder

In order to decode an LCU in any case, decoder needs to:

- Read bits with the worst case LCU size after compression, that is, **read 2x data of the original data size**;
- Read MC blocks associated with the worst case, 8x8 bi-pred, that is, **read 6.7x data of the original data size**.



Peak data transfer rates in different-resolution video decoding

- Bitstream read takes 20% of overall peak data transfer rate.
 - If a MC block cache is employed, the bitstream read takes more portions.

Video resolution	DRAM IO MB/s			
	Bitstream read	MC read	Picture write	Sum
720 30p	82.94	279.07	41.47	403.49
720 60p	165.89	558.14	82.94	806.98
1920 30p	188.01	632.56	94.00	914.57
1920 60p	376.01	1265.13	188.01	1829.15
4K2K 30p	758.94	2553.51	379.47	3691.92
4K2K 60p	1517.88	5107.02	758.94	7383.83

In order to reduce peak data transfer rates as much as possible, the peak bitstream read rates should be lowered by defining a lower expansion ratio, e.g., 1.33x or 1.60x.

Example of peak data rate reduction by different LCU bit size limits

2.00x limit case

Video resolution	DRAM IO MB/s			
	Bitstream read	MC read	Picture write	Sum
720 30p	82.94	279.07	41.47	403.49
720 60p	165.89	558.14	82.94	806.98
1920 30p	188.01	632.56	94.00	914.57
1920 60p	376.01	1265.13	188.01	1829.15
4K2K 30p	758.94	2553.51	379.47	3691.92
4K2K 60p	1517.88	5107.02	758.94	7383.83

1.60x limit case

Video resolution	DRAM IO MB/s				Rate reduction %
	Bitstream read	MC read	Picture write	Sum	
720 30p	66.36	279.07	41.47	386.90	4.11
720 60p	132.71	558.14	82.94	773.80	4.11
1920 30p	150.41	632.56	94.00	876.97	4.11
1920 60p	300.81	1265.13	188.01	1753.94	4.11
4K2K 30p	607.15	2553.51	379.47	3540.13	4.11
4K2K 60p	1214.30	5107.02	758.94	7080.26	4.11

1.33x limit case

Video resolution	DRAM IO MB/s				Rate reduction %
	Bitstream read	MC read	Picture write	Sum	
720 30p	55.16	279.07	41.47	375.70	6.89
720 60p	110.32	558.14	82.94	751.40	6.89
1920 30p	125.02	632.56	94.00	851.59	6.89
1920 60p	250.05	1265.13	188.01	1703.18	6.89
4K2K 30p	504.69	2553.51	379.47	3437.67	6.89
4K2K 60p	1009.39	5107.02	758.94	6875.34	6.89

Simulation for pseudo noise sequence

Limit of 1.33x expansion

- Sufficient margins for encoder to conform for QP=4.
- Reduce the bitstream read rates and the overall DRAM-decoder peak data transfer rates by 67% and 7%, respectively.

Limit of 1.60x expansion

- Sufficient margins for encoder to conform, even for QP=-8 in HE10.
- Reduce the bitstream read rates and the overall DRAM-decoder peak data transfer rates by 40% and 4%, respectively.

Table: Average expansion ratio for different LCU sizes

QP	Data expansion ratio for RA main			QP	Data expansion ratio for RA HE10		
	64x64 LCU	32x32 LCU	16x16 LCU		64x64 LCU	32x32 LCU	16x16 LCU
0	1.419	1.419	1.420	-12	1.732	1.731	1.736
4	1.264	1.264	1.265	-8	1.576	1.576	1.578
8	1.168	1.169	1.170	-4	1.410	1.410	1.412
12	1.023	1.024	1.023	0	1.151	1.151	1.152

Conclusions

- Study of the LCU bit size impact on peak data transfer rates between DRAM and decoder.
- LCU bit size limit of 1.33x expansion provides enough margins for encoder to conform while reducing the overall DRAM-decoder peak data transfer rates by 7%.
- Recommendations:
 - Fix the typos (“coding_tree()” and “coding tree unit”);
 - Consider the adoption of the 1.33x or 1.60x expansion limit in the spec. as below.

The number of times `read_bits(1)` is called in subclauses 9.3.3.2.2 and 9.3.3.2.3 when parsing `coding_quadtree()` data for any coding tree ~~unit block~~ shall not be greater than **512 (or 614) 768** * (`bit_depth_luma_minus8 + 8`) * (`1 << (Log2CtbSizeY - 4)`) * (`1 << (Log2CtbSizeY - 4)`).